INSTITUTO TECNOLÓGICO VALE

Programa de Pós-Graduação em Instrumentação, Controle e Automação de Processos de Mineração (PROFICAM)
Escola de Minas, Universidade Federal de Ouro Preto (UFOP)
Associação Instituto Tecnológico Vale (ITV)

Dissertação

PLANEJAMENTO DE CAMINHOS PARA ROBÔS MÓVEIS EM AMBIENTES ACIDENTADOS

Alexandre Souza Santos

Ouro Preto
Minas Gerais, Brasil
2019

Alexandre Souza Santos

PLANEJAMENTO DE CAMINHOS PARA ROBÔS MÓVEIS EM AMBIENTES ACIDENTADOS

Dissertação apresentada ao curso de Mestrado Profissional em Instrumentação, Controle e Automação de Processos de Mineração da Universidade Federal de Ouro Preto e do Instituto Tecnológico Vale, como parte dos requisitos para obtenção do título de Mestre em Engenharia de Controle e Automação.

Linha de Pesquisa: Robótica Aplicada à Mineração

Orientador: Prof. D.Sc. Gustavo Medeiros Freitas

Ouro Preto
2019

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

S237p Santos, Alexandre Souza .

Planejamento de caminhos para robôs móveis em ambientes acidentados. [manuscrito] / Alexandre Souza Santos. - 2019.

120 f.: il.: color., gráf., tab., mapa..

Orientador: Prof. Dr. Gustavo Medeiros Freitas.

Dissertação (Mestrado Profissional). Universidade Federal de Ouro Preto. Programa de Mestrado Profissional em Instrumentação, Controle e Automação de Processos de Mineração. Programa de Instrumentação, Controle e Automação de Processos de Mineração.

Área de Concentração: Engenharia de Controle e Automação de Processos Minerais.

1. Robôs - Sistemas de controle. 2. Equipamento agrícola. 3. Robótica. I. Santos, Alexandre Souza . II. Freitas, Gustavo Medeiros. III. Universidade Federal de Ouro Preto. IV. Título.

CDU 681.5:622.2

Bibliotecário(a) Responsável: Maristela Sanches Lima Mesquita - CRB: 1716

Mestrado Profissional em Instrumentação, Controle e Automação de Processos de Mineração - PROFICAM

Planejamento de Caminhos para Robôs Móveis em Ambientes Acidentados

Alexandre Souza Santos

Dissertação defendida e aprovada em 12 de julho de 2019 pela banca examinadora constituída pelos professores:

D.Sc. Gustavo Medeiros Freitas
Orientador – Universidade Federal de Minas Gerais (UFMG)

D.Sc. Armando Alves Neto
Membro externo – Universidade Federal de Minas Gerais (UFMG)

D.Sc. Wolmar Araújo Neto
Membro externo – Universidade Federal de Ouro Preto (UFOP)

D.Sc. Gustavo Pessin
Membro interno – Instituto Tecnológico Vale Mineração (ITV)

Dedico este trabalho à minha
família pelo apoio desde sempre.

Agradecimentos

Agradeço primeiramente aos meus pais, Luiz e Magda, pelo amor, carinho e apoio incondicional durante esse desafio chamado mestrado. Sem o suporte de vocês nada disso seria possível.

Agradeço aos meus irmãos, Luiz e Marina, tios, tias, avós e todos os familiares que sempre me apoiaram desde o início com palavras amigas, conselhos e também críticas construtivas.

Agradeço a todos os meus amigos, em especial, Jéssica, Halef, Eduardo, Kelvin, Arthur, Wallace, Weydster, Rafael, Ramon e Carlos, por todos esses anos de amizade e parceria.

Agradeço à UFOP e aos doutores pelo comprometimento com o ensino de qualidade durante o mestrado.

Agradeço aos meus orientadores Gustavo e Héctor por todo conhecimento passado e pela paciência.

Agradeço aos amigos do ITV, em especial, Marcos, Cristiano, Diego, Ênio, Cota, Jhonny, Aline, Felipe, Pedro, Sabrina, Thomas, Amaurir, Jacó, Pablo e Alexandre, que tiveram participação fundamental no desenvolvimento deste trabalho.

Agradeço aos amigos que fiz na República Taturrodano, em especial ao Cumade, Torresminho, Sem recheio, Tanajura, Minhoca e Jean BBB.

Agradeço a todos os amigos que fiz durante o mestrado, em especial os da segunda turma.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brasil (CAPES), Código de Financiamento 001; do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq); da Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG); e da Vale SA.

Resumo

Resumo da Dissertação apresentada ao Programa de Pós Graduação em Instrumentação, Controle e Automação de Processos de Mineração como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

PLANEJAMENTO DE CAMINHOS PARA ROBÔS MÓVEIS EM AMBIENTES ACIDENTADOS

Alexandre Souza Santos

Julho/2019

Orientador: Gustavo Medeiros Freitas

A equipe de Espeleologia da Vale possui uma plataforma robótica, o EspeleoRobô, capaz de se locomover em terrenos acidentados e realizar a inspeção e mapeamento de cavidades naturais. Por se tratar de um dispositivo teleoperado, sua operacionalidade depende da qualidade de comunicação com a base de controle. Uma solução para o problema seria dar autonomia ao EspeleoRobô. Sendo assim, o presente trabalho corresponde a um passo inicial para o desenvolvimento de um sistema de navegação autônomo, no qual o objetivo é analisar técnicas de planejamento de caminho que otimizem a navegação do robô em terrenos acidentados considerando seu modelo cinemático. O ambiente é representado por malhas triangulares modeladas como um grafo, onde são gerados caminhos ótimos utilizando o algoritmo de Dijkstra implementado no Matlab considerando as seguintes métricas como função de custo: distância percorrida, transversalidade do terreno e consumo energético do robô. Ainda, é proposta uma função de custo que considera múltiplos objetivos durante o planejamento de caminho, onde a relação entre elas é ajustada pelo operador por meio de pesos. São consideradas a arquitetura do robô e estimações dos pontos de contato do mesmo no ambiente, onde é realizado um estudo para analisar a interação do robô com o terreno utilizando os *softwares* Pybullet e V-REP. Para validação das métricas são feitas simulações utilizando mapas 3-D artificiais e um mapa real de cavidade natural.

Palavras-chave: Planejamento de Caminhos, Robótica Móvel, Robótica de Campo.

Macrotema: Instrumentação, Controle e Automação de Processos de Mineração; **Linha de Pesquisa:** Robótica Aplicada à Mineração; **Tema:** Robótica Móvel; **Área Relacionada da Vale:** Espeleologia e Tecnologia da Diretoria de Planejamento e Desenvolvimento de Ferrosos da Vale.

Abstract

Abstract of Dissertation presented to the Graduate Program on Instrumentation, Control and Automation of Mining Process as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

PATH PLANNING FOR MOBILE ROBOTS ON ROUGH TERRAIN

Alexandre Souza Santos

July/2019

Advisor: Gustavo Medeiros Freitas

Vale's Speleology team has a robotic platform, the EspeleoRobô, capable of moving on rough terrain while inspecting and mapping natural caves. Since the robot is a teleoperated device, its operability depends on the quality of communication with the control base. One solution to the problem is to provide autonomy for EspeleoRobô. Therefore, the present work consists on an initial step for the development of an autonomous navigation system, in which the objective is to analyze path planning techniques that optimize the robot navigation on rough terrain considering its kinematic model. We represent the environment using triangle meshes modeled as a graph and generate optimal paths with the Dijkstra's algorithm implemented in Matlab considering the following metrics as cost function: traveled distance, terrain traversability, and robot power consumption. Moreover, this work proposes a cost function that accounts for multiple goals during the path planning, where the user can set the trade-off between them through weights. We are considering the robot architecture and the contact points between the terrain and the robot, where we study the robot-terrain interaction using Pybullet and V-REP. We validate the metrics through simulations using artificial 3-D and real natural cave maps.

Keywords: Path Planning, Mobile Robotics, Field Robotics.

Macrotheme: Instrumentation, Control and Automation of Mining Processes; **Research Line:** Robotics Applied to Mining; **Theme:** Mobile Robotics; **Related Area of Vale:** Espeleologia e Tecnologia da Diretoria de Planejamento e Desenvolvimento de Ferrosos da Vale.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AT Ângulo de Tombamento

BFS *Breadth-First Search*

CIR Centro instantâneo de rotação do robô

DARPA *Defense Advanced Research Projects Agency*

DFS *Depth-First Search*

ITV-RoC Laboratório de Robótica e Controle do Instituto Tecnológico Vale - Ouro Preto/MG

PS Polígono Suporte

RRT *Rapidly-Exploring Random Trees*

SLAM *Simultaneous Localization and Mapping*

V-REP *Vehicle Readiness Enhancement Program*

Lista de Símbolos

α Fator de correção

A Matriz de modelagem cinemática do dispostivo *skid-steer* que pertence ao grupo $\mathbb{R}^{3x2}$

$AT_{ij}(n)$ Vetor com os seis ângulos de tombamentos do robô quando o centro de gravidade (CG) do mesmo está posicionado sobre a malha triangular de índice i com orientação em direção à malha de índice j

AT_{max} Maior ângulo de tombamento possível, isto é, quando o robô está em um plano sem inclinação

β_{ij} Rotação em graus que o robô precisa fazer entre os nós adjacentes de índices i e j

$C_1(p)$ Função de custo para a métrica de distância percorrida considerando o robô como uma partícula

$C_2(p)$ Função de custo para a métrica de transversalidade do terreno considerando o robô como uma partícula

$C_3(p)$ Função de custo para a métrica de consumo energético do robô considerando o mesmo como uma partícula

$C_4(p)$ Função de custo combinando as métricas de distância percorrida, transversalidade do terreno e consumo energético do robô considerando o mesmo como uma partícula

$C_5(p)$ Função de custo para a métrica de distância percorrida considerando a cinemática do robô

$C_6(p)$ Função de custo para a métrica de transversalidade do terreno considerando a cinemática do robô

$C_7(p)$ Função de custo para a métrica de energia obtida por meio de simulações com o V-REP

$C_8(p)$ Função de custo combinando as métricas de distância percorrida, transversalidade do terreno e consumo energétco do robô considerando a cinemática do mesmo

CG Centro de gravidade do robô

D Distância que se deseja mover um corpo em um plano inclinado

Dc_{ij} Métrica de distância percorrida considerando as rotações do robô entre os nós adjacentes de índices i e j

D_{ij} Distância euclidiana entre os nós adjacentes de índices i e j

d Distância percorrida pelo robô em linha reta

Ec_{ij} Energia gasta para rotacionar e deslocar o robô entre os nós adjacentes de índices i e j considerando a cinemática do robô

E_{ij} Energia gasta para deslocar o robô entre os nós adjacentes de índices i e j considerando o mesmo como uma partícula

$\vec{f_a}$ Força de atrito

$f(n)$ Função de custo

F_n Ponto de apoio de índice n

$\vec{f_p}$ Componente da força em um plano inclinado

g Aceleração da gravidade

$g(n)$ Custo para se deslocar da posição inicial até a posição atual

$h(n)$ Eurística que estima o custo de deslocamento da posição atual até a final

L Distância entre as rodas no mesmo eixo

l_5 Metade da distância entre as duas rodas centrais do EspeleoRobô

m Massa do robô

Nd_{ij} Coeficiente de normalização referente à métrica de distância percorrida

Ne_{ij} Coeficiente de normalização referente à métrica de consumo energético do robô

$\vec{N_i}$ Vetor normal da malha triangular de índice i

Nt_{ij} Coeficiente de normalização referente à métrica de transversalidade do terreno

ω Velocidade angular do robô

ω_L Velocidade angular da(s) roda(s) do lado esquerdo do robô

ω_R Velocidade angular da(s) roda(s) do lado direito do robô

p Caminho explorado pelo algoritmo

pc_{comb} Caminho de custo mínimo com as métricas combinadas considerando a cinemática do robô

pc_{dist} Caminho mais curto considerando a cinemática do robô

pc_{ener} Caminho mais econômico considerando a cinemática do robô

p_{comb} Caminho de custo mínimo com as métricas combinadas considerando o robô como uma partícula

pc_{tran} Caminho mais estável considerando a cinemática do robô

P_d Peso que dita a prioridade da métrica de distância percorrida

p_{dist} Caminho mais curto considerando o robô como uma partícula

P_e Peso que dita a prioridade da métrica de consumo energético do robô

p_{ener} Caminho mais econômico considerando o robô como uma partícula

P_t Peso que dita a prioridade da métrica de transversalidade do terreno

p_{tran} Caminho mais plano considerando o robô como uma partícula

r Raio da roda

$R_z(\theta)$ Matriz de rotação elementar no eixo z pertencente ao grupo ortogonal $SO(3)$

Tc_{ij} Métrica de transversalidade do terreno considerando a cinemática do robô

θ^* Ângulo obtido após uma sequência de rotações em torno do próprio eixo

$\dot{\theta}$ Velocidade angular do robô

θ_{ij} Ângulo formado entre o vetor que liga os centros das malhas i e j e o plano horizontal definidos pelos eixos X e Y do terreno

Θ_n Vetor com os ângulos necessários para levar o dispostivo até o limiar da estabilidade sobre as n arestas de sua borda de suporte

T_i Ângulo agudo positivo entre a normal ($\vec{N}_i$) da malha triangular de índice i e o eixo Z

μ Coeficiente de atrito do terreno

v Velocidade linear do robô

V_L Velocidade linear da(s) roda(s) do lado esquerdo do robô

V_R Velocidade linear da(s) roda(s) do lado direito do robô

$\vec{X}$ Eixo X

x_{CIR} Centro instantâneo de rotação

$\dot{x}$ Velocidade linear do robô em relação ao eixo X

$\vec{Y}$ Eixo Y

$\dot{y}$ Velocidade linear do robô em relação ao eixo Y

$\vec{Z}$ Eixo Z

Sumário

Capítulo 1

Introdução

Um dos maiores desafios da robótica móvel é desenvolver robôs capazes de realizar tarefas autônomas. Esses robôs podem ser utilizados em diversas aplicações como de inspeção e manutenção (SCHMIDT e BERNS, 2013), vigilância (ACEVEDO *et al.*, 2013), transporte urbano (KÜMMERLE *et al.*, 2013), segurança e defesa (RABOIN *et al.*, 2013), limpeza (ZIEGLER *et al.*, 2013), entre outras.

Robôs móveis também estão sendo empregados para explorar ambientes desconhecidos ou com riscos para a vida humana, por exemplo exploração planetária (SCHUSTER *et al.*, 2019). Outro exemplo onde esses robôs estão sendo utilizados é no campo da mineração. Empresas que trabalham nessa área precisam inspecionar regiões com presença de cavidades naturais para determinar a relevância da cavidade no processo de viabilização de um projeto de mineração.

Para viabilizar as operações nestas regiões, a equipe de espeleologia da empresa deve visitar todas as cavidades para realizar um mapeamento e coletar dados que servirão de base para os estudos ambientais exigidos. No entanto, a maioria das cavidades em regiões com presença de minério de ferro são estreitas e de difícil acesso, onde o profissional em trabalho de campo pode encontrar vários riscos como desabamentos e presença de animais selvagens e peçonhentos (Figura 1.1).

Figura 1.1: Riscos relacionados à inspeção de cavidades naturais: desabamentos e presença de animais selvagens e peçonhentos (Arquivo público da internet).

Com intuito de auxiliar o trabalho do espeleólogo, evitando sua presença nesses locais, a Equipe de Espeleologia e Tecnologia da Diretoria de Planejamento e Desenvolvimento de Ferrosos da Vale deu início ao projeto do EspeleoRobô, adquirindo uma plataforma robótica móvel teleoperada capaz de se locomover em terrenos acidentados e realizar a inspeção de cavidades. Essa plataforma possui uma arquitetura similar ao robô RHex (SARANLI *et al.*, 2001), que utiliza seis pernas para se locomover e ultrapassar obstáculos em ambientes não estruturados (Figura 1.2).

(a) (b)

Figura 1.2: (a) RHex da *Boston Dynamics*; (b) EspeleoRobô da Vale.

O primeiro protótipo adquirido pela equipe apresentava problemas de sincronização, além de carência tecnológica. Devido a isso, o dispositivo foi enviado ao Laboratório de Robótica e Controle do Instituto Tecnológico Vale - Ouro Preto/MG (ITV-RoC) para aprimoramentos. Em pouco tempo de trabalho o ITV realizou melhorias mecânicas, de *hardware* e de *software*, onde quase todos os componentes foram substituídos (COTA *et al.*, 2017).

1.1 Motivação

Atualmente, a transmissão de sinais entre o EspeleoRobô e a base de controle é feita via rádio modelo Ubiquiti Rocket M900, que possui frequência de operação de 900MHz e trabalha com potência de saída de até 28dBm. A base conta com uma antena direcional Yagi AMY-9M16 de ganho 16dBi e o robô possui duas antenas omnidirecionais ASA-900CI de ganho 8,15dBi, que trabalham com a mesma faixa de operação. Apesar dos êxitos em diversos testes realizados em campo, foram constatados problemas recorrentes de perda de comunicação entre a base e o robô. Isso porque a comunicação via rádio apresenta limitações quando se trata de ambientes não-estruturados com obstáculos como cavidades. Essas falhas podem causar danos na estrutura do robô caso esse caia em um buraco ou até mesmo a perda do equipamento caso esse entre em uma área que impossibilite o resgate.

Uma solução para o problema seria dar autonomia ao EspeleoRobô. Dessa maneira, um sistema de instrumentação e controle embarcados tomaria as decisões de forma a aprimorar a exploração e garantir a integridade do equipamento. O robô conta com uma torre de mapeamento composta por câmeras de alta resolução e o sensor Velodyne LiDAR VLP-16. A modelagem tridimensional da torre de instrumentação acoplada sobre o robô e um exemplo de teste em campo estão ilustrados na Figura 1.3.

(a) (b)

Figura 1.3: (a) Modelagem tridimensional da torre de instrumentação; (b) exemplo de teste em campo. Fonte: Rocha *et al.* (2017).

Para ressaltar a importância do tema desse trabalho, a Agência de Projetos de Pesquisa Avançada de Defesa dos Estados Unidos (DARPA) lançou recentemente um desafio para a comunidade científica para estimular o desenvolvimento de tecnologias para o subsolo. O desafio consiste em três etapas, na qual uma delas está relacionada com o mapeamento e navegação em ambientes como cavidades naturais (DARPA, 2017). Sendo assim, nos próximos anos a comunidade cientifica deve se interessar por esse tema.

1.2 Objetivos

Os objetivos estão divididos em geral e específicos. O objetivo geral apresenta a ideia central a ser desenvolvida pelo trabalho, e os específicos são os desdobramentos necessários para a alcançar a meta principal.

1.2.1 Gerais

Esta dissertação tem como objetivo analisar técnicas de planejamento de caminho afim de otimizar a exploração do robô em ambientes acidentados.

Para validação das técnicas são utilizados ambientes representados por meio de malhas triangulares modeladas como um grafo, onde são gerados caminhos ótimos utilizando o algoritmo de Dijkstra implementado no Matlab (HANSELMAN e LITTLEFIELD, 2001) considerando as seguintes métricas como função de custo: distância percorrida, transversalidade do terreno e consumo energético do robô.

Nesta dissertação o planejamento de caminhos é realizado em duas etapas. A primeira etapa considera o robô como uma partícula, na qual a métrica de distância é calculada pela soma das distância euclidianas entre pontos que o robô percorre no terreno. A métrica de transversalidade é calculada pela soma dos ângulos da malhas do terreno percorridas pelo robô. A métrica de energia é calculada pela soma dos trabalhos necessários para mover um corpo em planos inclinados. Os resultados dessa etapa foram publicados em Santos *et al.* (2018).

A segunda etapa do planejamento considera a arquitetura do robô. Nesse caso a métrica de distância incorpora o giro das rodas necessário para realizar deslocamentos lineares e angulares. De maneira análoga, a métrica de consumo energético para essa etapa considera a energia que o robô gasta para rotacionar, além da energia gasta em movimentos lineares. A métrica de transversalidade considera a arquitetura do robô para avaliar a interação do mesmo com o terreno, calculando os ângulos de tombamento do dispositivo com base nas posições dos pontos de contato entre o robô e o terreno.

Também é proposta uma função de custo que considera múltiplas métricas durante o planejamento de caminho. A relação entre as métricas é ajustada por meio de pesos definidos pelo operador. Dessa forma, o algoritmo de planejamento de caminho avalia a relevância de cada métrica ao encontrar trajetos ligando as posições inicial e final. Por fim, é realizado um estudo para analisar a interação do robô com o terreno utilizando os *softwares* Pybullet (COUMANS e BAI, 2016) e *Vehicle Readiness Enhancement Program* V-REP (ROHMER *et al.*, 2013). Assim, o algoritmo tenta prevenir o tombamento do robô, garantindo sua integridade durante a exploração.

1.2.2 Específicos

De maneira mais direta, são desenvolvidas as seguintes atividades:

- Definir as melhores técnicas de planejamento de caminho para ambientes acidentados;
- Implementar e validar as técnicas de planejamento de caminho utilizando o *software* Matlab;
- Analisar caminhos obtidos por meio de otimização das métricas de distância percorrida, transversalidade do terreno e consumo energético do robô;

- Propor um método para gerar caminhos que combinem as diferentes métricas;
- Estudar a interação do robô com o terreno utilizando os *softwares* Pybullet e V-REP;
- Analisar os resultados das simulações.

1.3 Contribuições da dissertação

O projeto EspeleoRobô conta com diversas atividades realizadas de forma paralela, como o desenvolvimento de bibliotecas de comunicação, mapeamento tridimensional, instalações elétricas, dentre outras, ilustradas pela Figura 1.4. E o problema de navegação autônoma envolve várias tarefas como mapeamento, localização, planejamento de caminho e controle da trajetória (Figura 1.5). As contribuições desta dissertação estão relacionadas ao enriquecimento do conteúdo bibliográfico que diz respeito ao planejamento de caminhos para robôs móveis em ambientes acidentados, e às metodologias utilizadas, testadas e validadas no presente trabalho. São utilizados diferentes *softwares* no processo de desenvolvimento e validação do algoritmo de planejamento de caminho como o Matlab, Pybullet e V-REP. As métricas de transversalidade do terreno e de consumo energético do robô utilizam valores obtidos por meio de simulações realísticas, o que torna as métricas mais robustas. Ainda, essa dissertação representa um estudo inicial para o desenvolvimento de sistema de navegação autônomo para o EspeleoRobô.

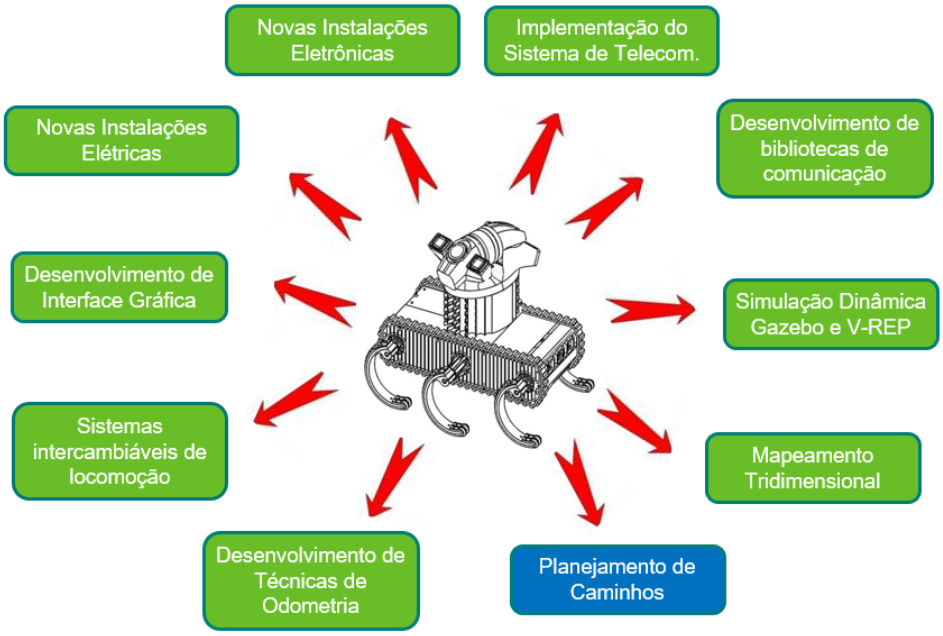


Figura 1.4: Atividades relacionadas ao projeto EspeleoRobô.

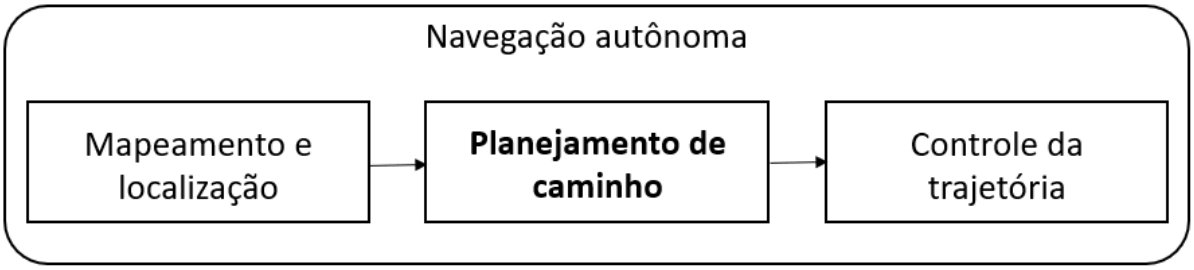


Figura 1.5: Representação das tarefas necessárias para navegação autônoma.

No desenvolvimento desta dissertação o autor contou com o apoio essencial dos orientadores, pesquisadores, mestrandos e bolsistas de graduação do ITV, que contribuíram de maneira fundamental para a obtenção dos resultados apresentados aqui.

1.4 Estrutura da dissertação

Esta dissertação é composta por 6 capítulos. No Capítulo 1, Introdução, é apresentada uma breve introdução sobre o problema abordado, assim como a motivação e os objetivos. No Capítulo 2, Revisão bibliográfica, são expostos os conceitos teóricos como fundamentação básica para o estudo. No Capítulo 3, Método proposto, é apresentada a metodologia utilizada para o desenvolvimento do trabalho. No Capítulo 4, Resultados e discussões, são apontados os resultados e as análises obtidos por meio de simulações. Por fim, no Capítulo 5, Conclusões e trabalhos futuros, são apresentadas as conclusões e contribuições da dissertação, e sugestões de atividades futuras.

Capítulo 2

Revisão bibliográfica

Neste capítulo são expostos os conceitos teóricos como fundamentação básica para o estudo. Ainda, é feito um breve resumo sobre trabalhos relacionados ao planejamento de caminho para robôs móveis.

2.1 Planejamento de caminho para robôs móveis

O processo de planejamento de caminho para robôs móveis vem sendo estudado há décadas (GONZÁLEZ *et al.*, 2015). Esse processo pode ser dividido basicamente em duas etapas. A primeira etapa consiste na representação do ambiente de navegação. Essa etapa é geralmente realizada por meio de sensores como câmeras, *lasers*, sonares, entre outros.

A segunda etapa consiste na utilização de algoritmos de busca para encontrar um caminho ótimo para o robô entre as posições inicial e final definidas pelo usuário. Diferentes trajetos podem ser obtidos de acordo com o critério de busca, como menor distância percorrida ou menor consumo energético (GIESBRECHT, 2004).

Os modelos de representação do ambiente podem ser divididos em duas categorias: modelos para ambientes fechados e estruturados, e modelos para ambientes naturais.

Os principais modelos para ambientes fechados e estruturados são: ocupação de grades, mapas de linha, mapas topológicos e mapas baseados em pontos de referências. Já os principais modelos para ambientes naturais são: grades de elevação, grades 3-D, malhas, mapas de custo, mapas com atributos semânticos e mapas hierárquicos e heterogêneos (BURGARD *et al.*, 2016).

Nesta dissertação são estudados os principais modelos geométricos para representação de ambientes naturais e os algoritmos de busca compatíveis com esse tipo de representação.

2.1.1 Modelos para representação de ambientes naturais

Segundo Burgard *et al.* (2016), os principais modelos geométricos para representação de ambientes naturais são grades de elevação, grades 3-D e malhas.

Os modelos de grades de elevação descrevem o terreno como uma função $h = f(x, y)$, onde x e y representam as coordenadas planares das células da grade e h representa a elevação correspondente. Devido às suas características de estrutura de dados simples e de fácil obtenção, mapas de elevação têm sido bastante utilizados em diversas aplicações no ramo da robótica móvel (BROGGI *et al.*, 2013; FANKHAUSER *et al.*, 2014). Um exemplo dessa representação está ilustrado pela Figura 2.1.

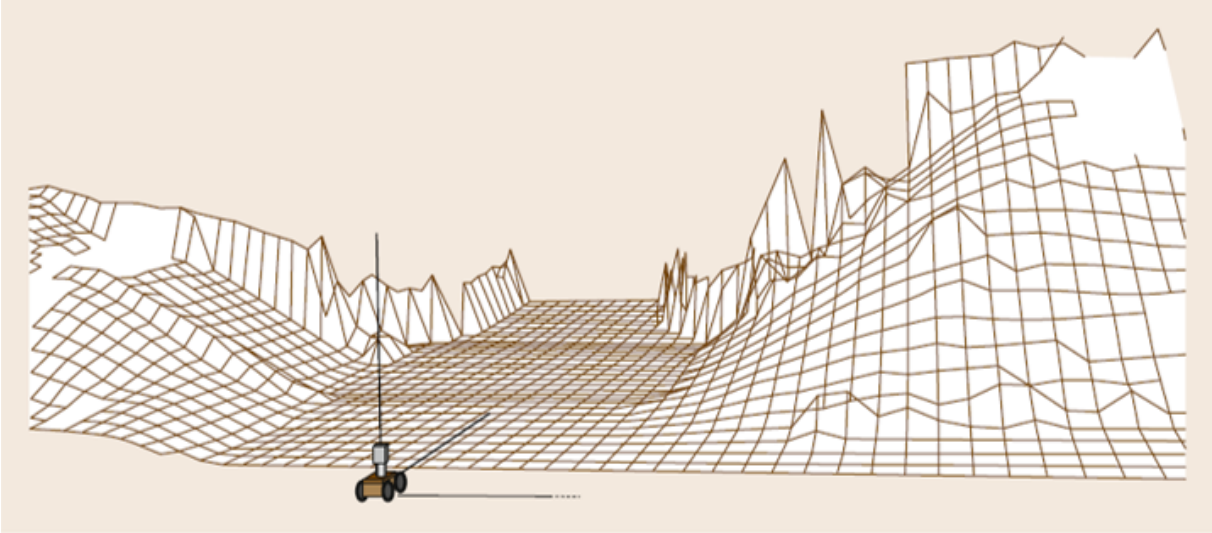


Figura 2.1: Mapa de elevação construído por meio de dados de um sensor de proximidade. Fonte: Burgard *et al.* (2016).

Grades de elevação assumem um plano como referência. No entanto, essa suposição é violada em diversas situações, por exemplo em casos onde a mesma coordenada (x, y) é associada a diferentes elevações h, inviabilizando a utilização desse tipo de modelo. Uma solução consiste em representar o terreno diretamente em nuvem de pontos ou grades tridimensionais (HORNUNG *et al.*, 2013). Dessa forma, todos os dados são preservados em seu formato original, não havendo restrição de geometria do ambiente. Um exemplo desse tipo de modelo está representado pela Figura 2.2.

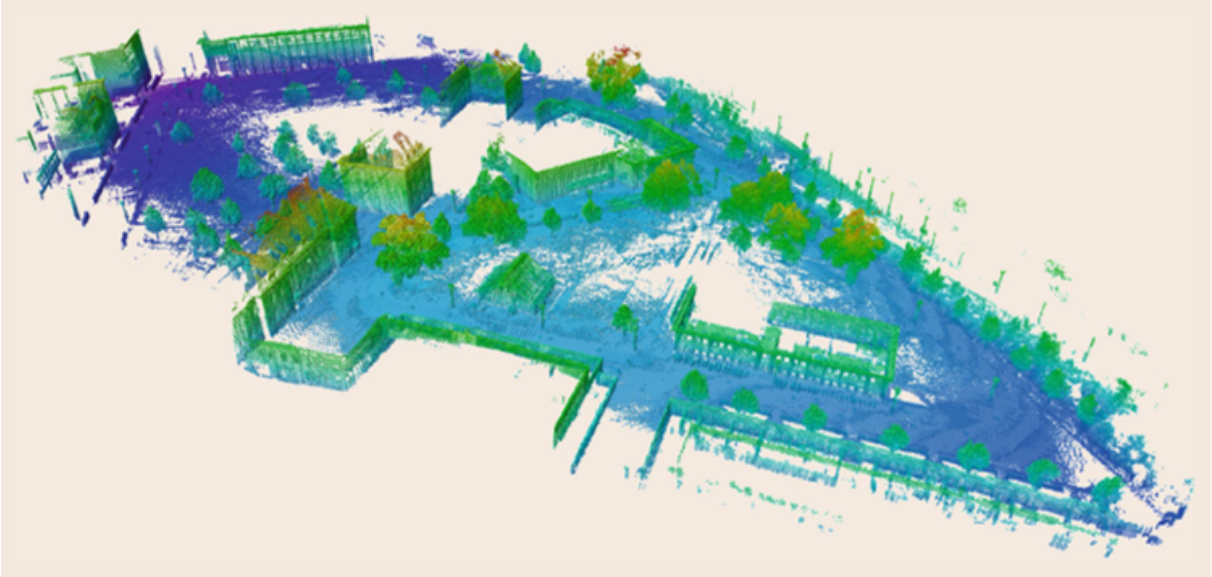


Figura 2.2: Ambiente ao ar livre representado por grades 3-D. As cores do mapa estão relacionadas com o eixo Z. Fonte: Burgard *et al.* (2016).

Apesar de ser uma representação mais genérica, as grades 3-D têm um custo computacional elevado, tornando-as inviáveis em determinadas aplicações. Outra alternativa é representar o terreno por meio de modelos de malhas (LIPMAN, 2012). Esses modelos podem representar qualquer combinação de superfícies de uma forma compacta. Contudo, esse tipo de representação pode encontrar dificuldades quando se trata de ambientes muito complexos. Um exemplo dessa representação está ilustrado pela Figura 2.3.

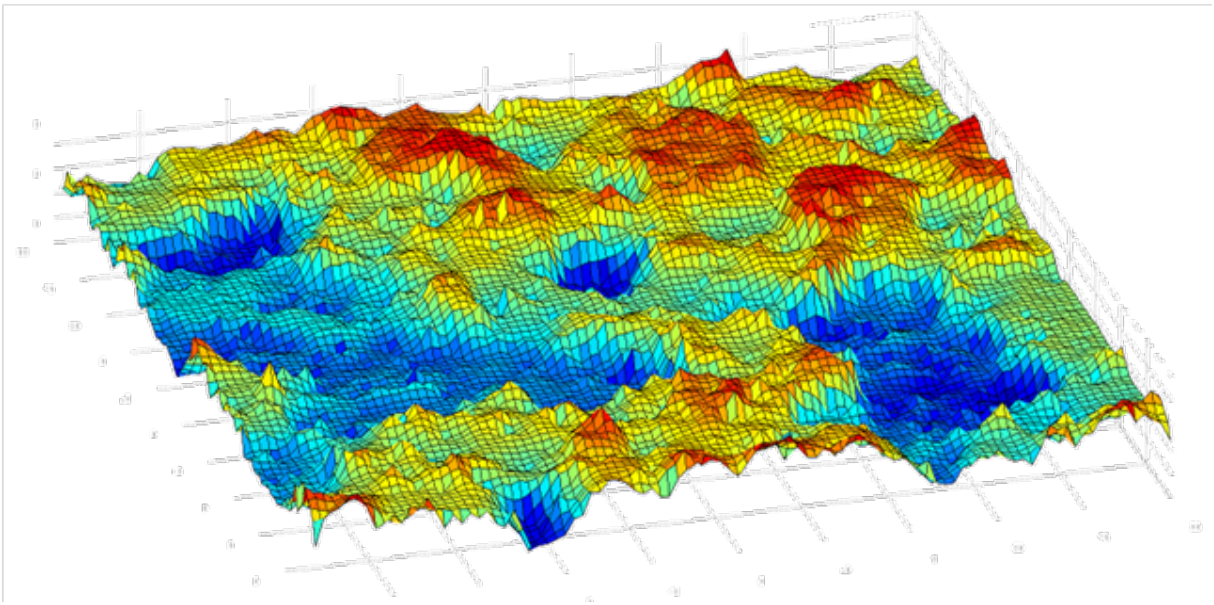


Figura 2.3: Mapa 3-D de um terreno acidentado representado por malhas retangulares. As cores do mapa estão relacionadas com o eixo Z. Fonte: Brown (2012).

2.1.2 Algoritmos de busca de caminho

Uma vez que o ambiente foi representado utilizando algum dos modelos geométricos citados na Seção 2.1.1 que podem ser representados como grafos, é necessário buscar o melhor caminho entre as posições inicial e final, sendo ambas definidas pelo usuário.

Grafos são estruturas compostas por nós ou vértices ligados por arcos ou linhas, onde pode-se atribuir pesos aos arcos. Dessa forma, um algoritmo de busca em grafo pode ser utilizado para encontrar o caminho com menor custo entre dois nós. Os principais algoritmos dessa categoria que solucionam problemas com grafos com arcos de pesos iguais são: busca em profundidade ou *Depth-First Search* (DFS), busca em largura ou *Breadth-First Search* (BFS) e árvores aleatórias para exploração rápida ou *Rapidly-Exploring Random Trees* (RRT). Os algoritmos A* e Dijkstra são utilizados em problemas de busca em grafos com arcos de pesos fixos, distintos e não-negativos, enquanto o D* é utilizado em problemas onde os pesos podem ser variáveis (GIESBRECHT, 2004).

O algoritmo de busca em profundidade, inicialmente proposto pelo matemático francês Charles Pierre Trémaux em 1876, tenta encontrar um caminho entre as posições inicial e final de um grafo com arcos de pesos iguais da maneira mais rápida possível (TRÉMAUX, 2010). Esse algoritmo realiza uma busca não-informada, ou seja, não utiliza informação adicional sobre os nós não explorados, continuando em um mesmo trajeto até encontrar o objetivo. Caso esse caminho não leve ao destino, a busca retrocede e começa em um outro percurso. Em analogia com uma árvore de busca, o algoritmo progride por meio da expansão do primeiro nó filho da árvore e se aprofunda até que o alvo da busca seja encontrado ou até que ele se depare com um nó que não possui filhos. Então a busca retrocede e começa no próximo nó. Uma representação desse algoritmo está ilustrada na Figura 2.4. Isso significa que esse método não explora os caminhos (nós) de forma simultânea, preferindo escolher o percurso mais curto até a posição final. Esse algoritmo é indicado para problemas pequenos com múltiplas soluções, onde apenas uma delas é necessária. Em problemas muito grandes a busca em profundidade não converge, pois o comprimento de um caminho solução tende ao infinito (OTTONI, 2000).

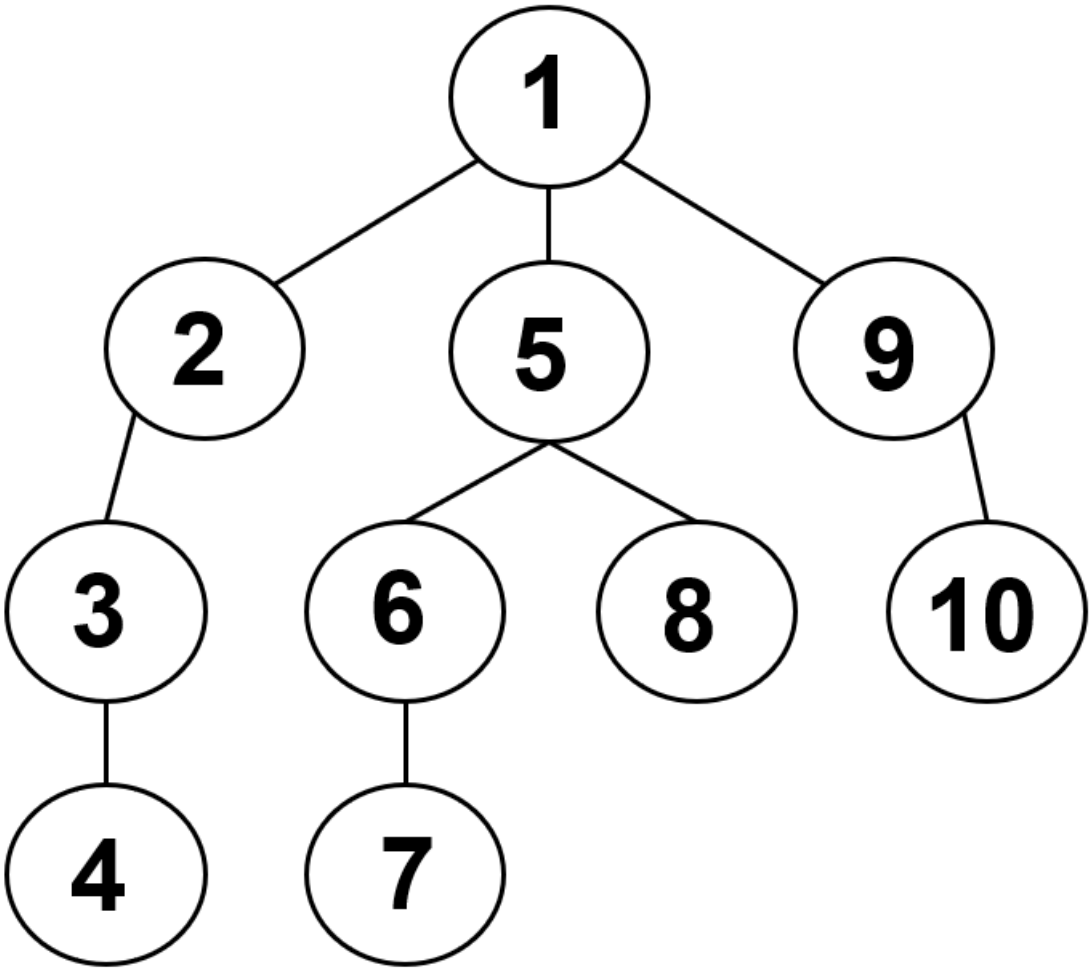


Figura 2.4: Representação do algoritmo busca em profundidade.

O algoritmo de busca em largura, proposto pelo engenheiro civil alemão Konrad Zuse em 1945, também procura a posição final de um grafo com arcos de pesos iguais. No entanto, diferente da busca em profundidade, a busca em largura analisa os vizinhos de primeiro grau. Em outras palavras, todos os sucessores de um caminho são analisados até o que algoritmo avance na busca. Em analogia com uma árvore de busca, o algoritmo progride de forma a explorar todos os filhos de um mesmo nó antes de prosseguir para a próxima geração. Assim, os percursos são avaliados simultaneamente, garantido que todos serão explorados e que o trajeto mais curto será encontrado. Esse algoritmo é apropriado para problemas relativamente pequenos com poucas soluções (GIESBRECHT, 2004). Uma representação desse algoritmo está ilustrada na Figura 2.5.

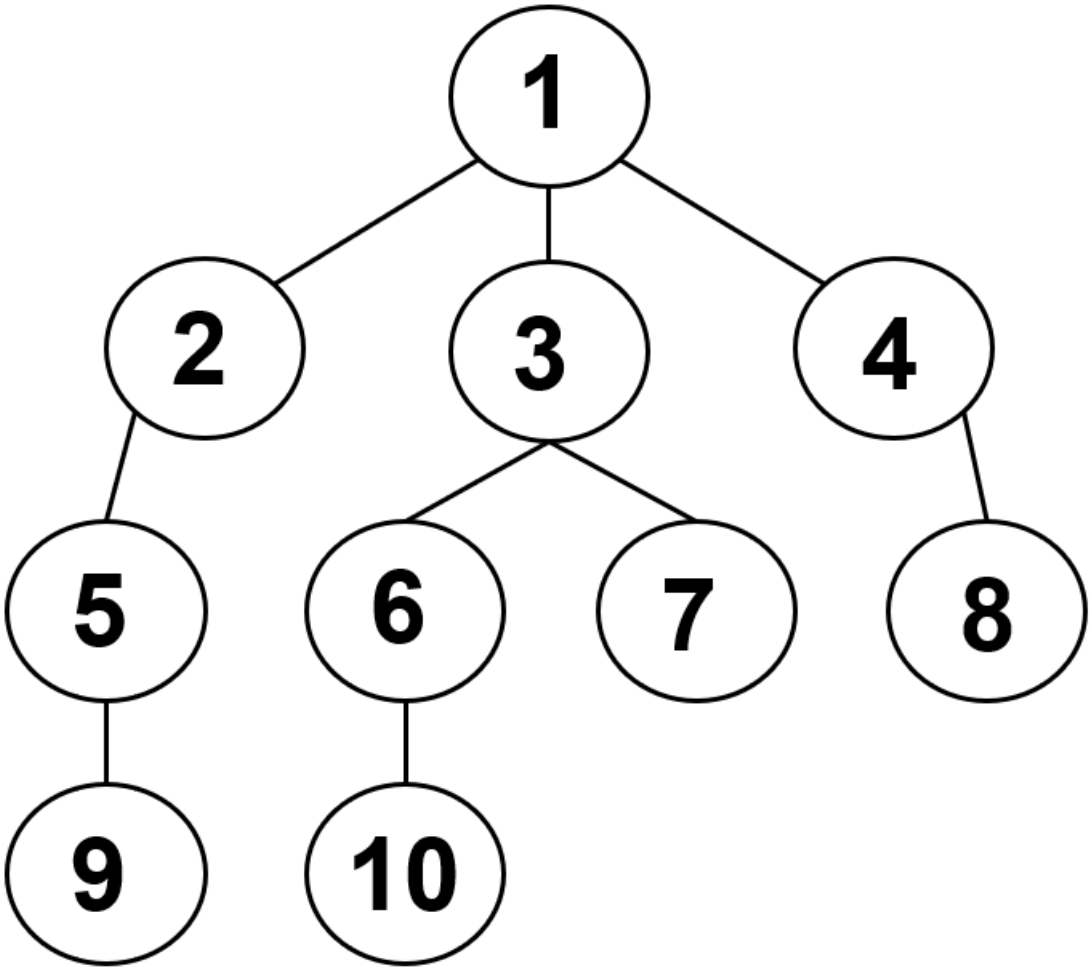


Figura 2.5: Representação do algoritmo busca em largura.

O algoritmo de árvore aleatória para exploração rápida possui uma abordagem um pouco diferente dos algoritmos supracitados (LAVALLE, 1998). Esse algoritmo constrói uma árvore de forma a ligar as posições inicial e final por meio de arcos. Esses arcos expandem-se em incrementos lineares das posições inicial e final de forma a cobrir o espaço livre do ambiente. Quando os dois caminhos estão próximos um do outro, eles são conetados gerando uma solução (Figura 2.6). Esse algoritmo é bastante utilizado em problemas com obstáculos e restrições dinâmicas não-holonômicas e kinodinâmicas (GIESBRECHT, 2004).

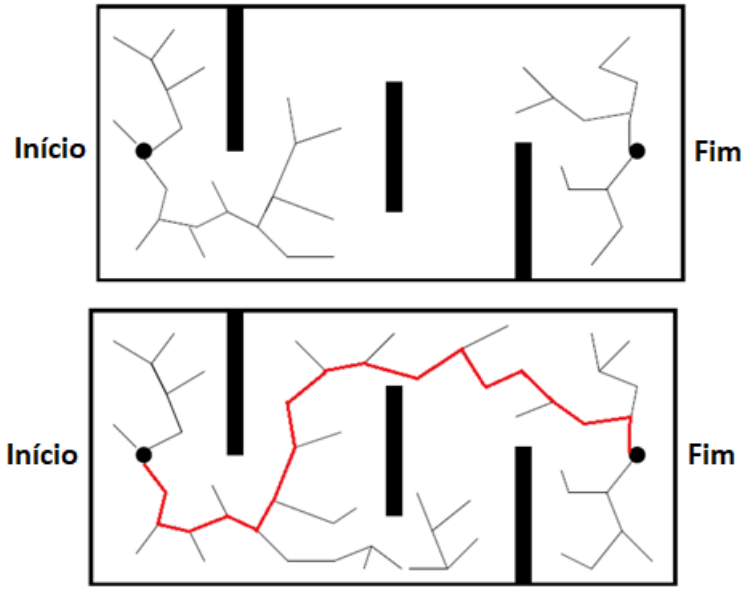


Figura 2.6: Representação do algoritmo RRT. Fonte: Giesbrecht (2004).

O algoritmo A*, inicialmente proposto por Hart *et al.* (1968) é um dos mais utilizados em aplicações de robótica. Esse método de busca utiliza a função de custo $f(n) = g(n) + h(n)$ para orientar a pesquisa do caminho ótimo, sendo $g(n)$ o custo para se deslocar da posição inicial até a posição atual e $h(n)$ é a heurística que estima o custo de deslocamento da posição atual até a final. Os arcos dos grafo possuem pesos fixos associados aos custos calculados. Dessa maneira, o algoritmo escolhe sempre o sucessor com o menor valor. O algoritmo A* apresenta algumas características que o torna tão famoso. Esse método de busca utiliza uma heurística admissível. Isso significa que o A* subestima o custo $h(n)^*$ do caminho ótimo, ou seja, $h(n) \leq h(n)^*$. Isso faz com que o algoritmo explore outros trajetos com custo inferior ao percurso ótimo, encerrando a busca somente quando o caminho mais curto ou ótimo é encontrado. Ainda, esse algoritmo apresenta a busca mais rápida se comparado com outros algoritmos que utilizam a mesma heurística para encontrar o trajeto mais curto (GIESBRECHT, 2004). Uma ilustração do algoritmo está representada pela Figura 2.7.

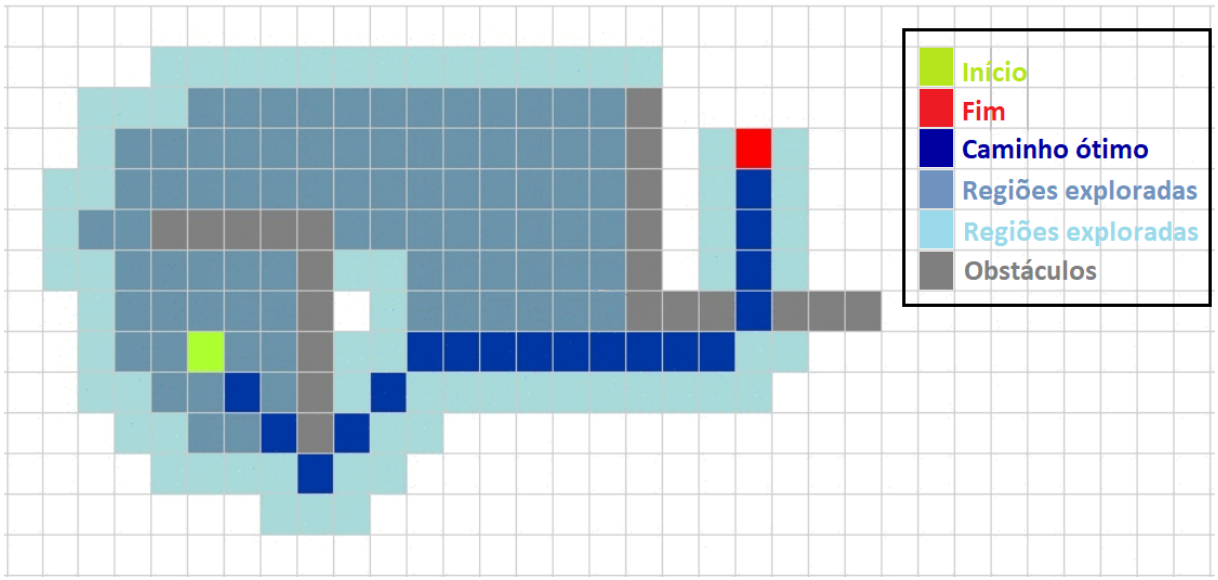


Figura 2.7: Representação do algoritmo A*. Modificado de Vittin (2017).

O algoritmo de Dijkstra (DIJKSTRA, 1959) é uma variação do algoritmo A* onde $h(n) = 0$, ou seja, $f(n) = g(n)$. Dessa forma, esse método não utiliza nenhuma heurística na busca do percurso ótimo, fazendo somente a expansão do caminho mais curto até que a posição final seja encontrada (CHOSET, 2005). O algoritmo de Dijkstra é utilizado para encontrar o caminho de custo mínimo entre vértices de um grafo com arcos de peso não-negativo. Para problemas em grafos com arestas de peso negativo, o algoritmo de Bellman-Ford (BELLMAN, 1958) pode ser empregado. Escolhido um vértice como raiz da busca, o algoritmo de Dijkstra calcula o custo mínimo desse vértice para todos os demais vértices do grafo. O algoritmo de Dijkstra parte de uma estimativa inicial para o custo mínimo e vai sucessivamente ajustando essa estimativa. Um vértice é dito estar fechado quando o algoritmo já tiver obtido um caminho de custo mínimo do vértice tomado como raiz da busca até ele. Caso contrário ele é dito estar aberto. Quando todos os vértices tiverem sido fechados, os valores obtidos são os custos mínimos dos caminhos que partem do vértice raiz até os demais vértices do grafo. O percurso de custo mínimo propriamente dito é obtido a partir dos vértices chamados de precedentes. Um exemplo de como esse algoritmo funciona está ilustrado pela Figura 2.8.

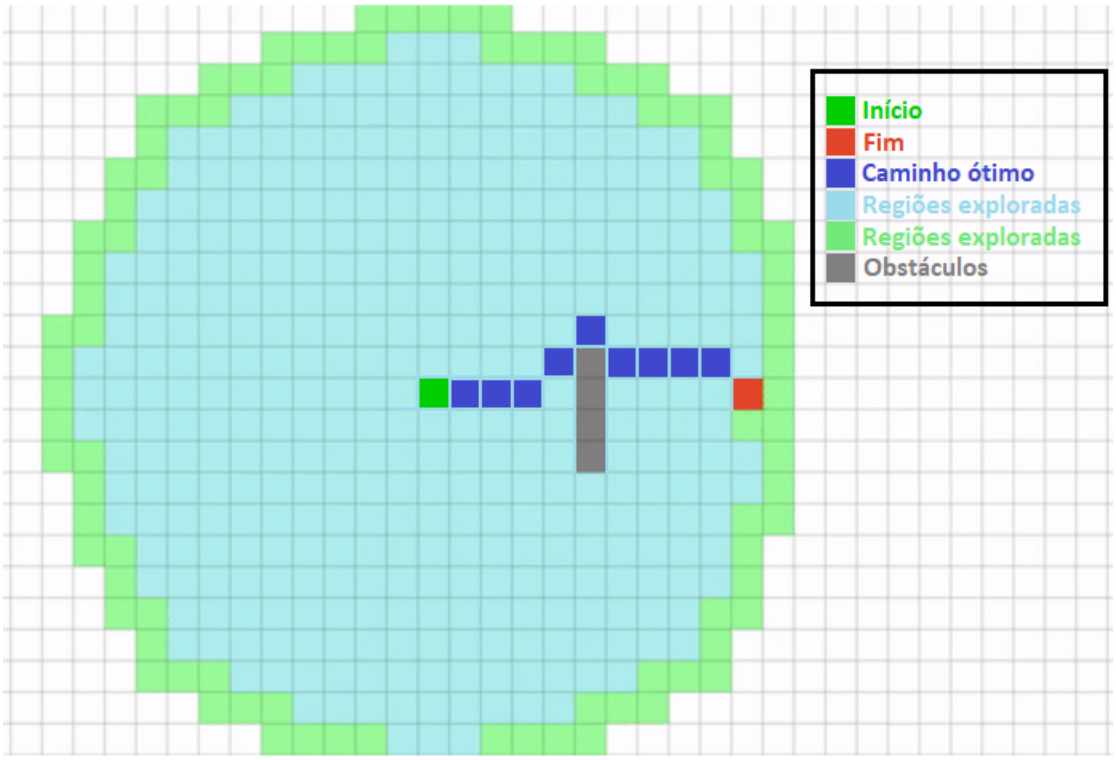


Figura 2.8: Representação do algoritmo de Dijkstra. Modificado de Holczer (2018).

Os algoritmos mencionados anteriormente dependem que o ambiente representado seja totalmente conhecido. Caso contrário, o algoritmo D* (STENTZ, 1994) pode ser empregado. Esse método de busca é utilizado para ambientes parcialmente conhecidos, estáticos ou dinâmicos. O D* é baseado nas ideias do A*, a diferença é que o A* utiliza custos fixos para os arcos do mapa e apenas calcula o caminho ótimo entre as posições inicial e final, enquanto no D* os custos dos arcos podem mudar de acordo com o ambiente e são calculados todos os possíveis trajetos entre as posições inicial e final, tornando-o muito custoso computacionalmente. No entanto, suas vantagens podem ser vistas em ambientes dinâmicos de grandes dimensões onde são necessárias constantes alterações nas rotas (GIESBRECHT, 2004).

2.2 Robós móveis

Nos dias atuais existe uma grande variedade de robôs móveis, no qual a maioria possui características particulares que os tornam aptos para determinadas tarefas. Essas tarefas determinam as características na estrutura de um robô como o tipo de roda, sistema de tração e direção e a forma física do robô (SECCHI, 2008). Nesta seção é apresentada uma breve revisão sobre os principais modelos de robôs móveis com rodas: uniciclo, diferencial e *skid-steer*.

2.2.1 Modelo uniciclo

O modelo do uniciclo possui a estrutura mais simples dentre os robôs com rodas. O modelo é basicamente uma roda que possui uma velocidade linear v e uma velocidade angular ω, e está representado pela Figura 2.9 (FRANCIS e MAGGIORE, 2016). Para representar esse modelo são assumidas as seguintes simplificações:

- o robô se move em uma superfície plana;
- não há deslizamento;
- não há deformação nas rodas;
- os eixos de orientação são perpendiculares ao solo.

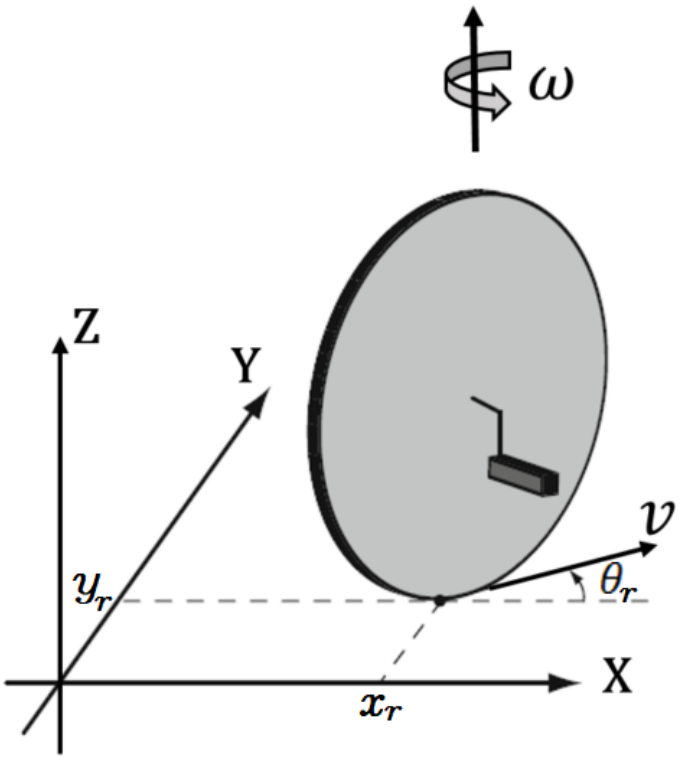


Figura 2.9: Modelo geométrico do uniciclo. Modificado de Francis e Maggiore (2016).

Considerando o modelo planar, tem-se que a pose do robô é dada por:

$$q = \begin{bmatrix} x_r \\ y_r \\ \theta_r \end{bmatrix}. \tag{2.1}$$

onde x_r e y_r são as componentes da posição nos eixos ortogonais X ($\vec{X}$) e Y ($\vec{Y}$), e θ_r é o ângulo de orientação em torno do eixo Z ($\vec{Z}$). A relação entre a posição (x_r, y_r) e

orientação θ_r do uniciclo em função da suas velocidades linear v e angular ω pode ser descrita da seguinte maneira:

$$\begin{bmatrix} \dot{x_r} \\ \dot{y_r} \\ \dot{\theta_r} \end{bmatrix} = \begin{bmatrix} vcos(\theta_r) \\ vsen(\theta_r) \\ \omega \end{bmatrix}. \tag{2.2}$$

Em geral, a cinemática de um robô móvel pode ser expressada na forma de modelo cinemático, onde $\dot{q}$ são as variáveis de estados, $\dot{\theta}$ são as variáveis de atuação e J(θ) é a matriz Jacobiana:

$$\dot{q} = J(\theta)\dot{\theta}. \tag{2.3}$$

Relacionando as Equações 2.2 e 2.3, tem-se que:

$$\begin{bmatrix} \dot{x_r} \\ \dot{y_r} \\ \dot{\theta_r} \end{bmatrix} = \begin{bmatrix} vcos(\theta_r) & 0 \\ vsen(\theta_r) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{2.4}$$

2.2.2 Modelo diferencial

O sistema de direção diferencial é um dos mais simples para um robô autônomo terrestre. Ele consiste basicamente em duas rodas montadas num eixo comum e controladas por motores independentes. Para que o robô gire em torno do Centro Instantâneo de Rotação (CIR) que está na direção do eixo comum às duas rodas, ilustrado pela Figura 2.10, é necessário variar a velocidade relativa entre as duas rodas. O ponto de giro pode ser alterado, fazendo com que o robô possa realizar trajetórias diferentes. Em cada instante, o ponto de giro do robô tem a propriedade de que as rodas direita e esquerda seguem caminhos que se movem em torno do CIR à mesma velocidade angular (DUDEK e JENKIN, 2010).

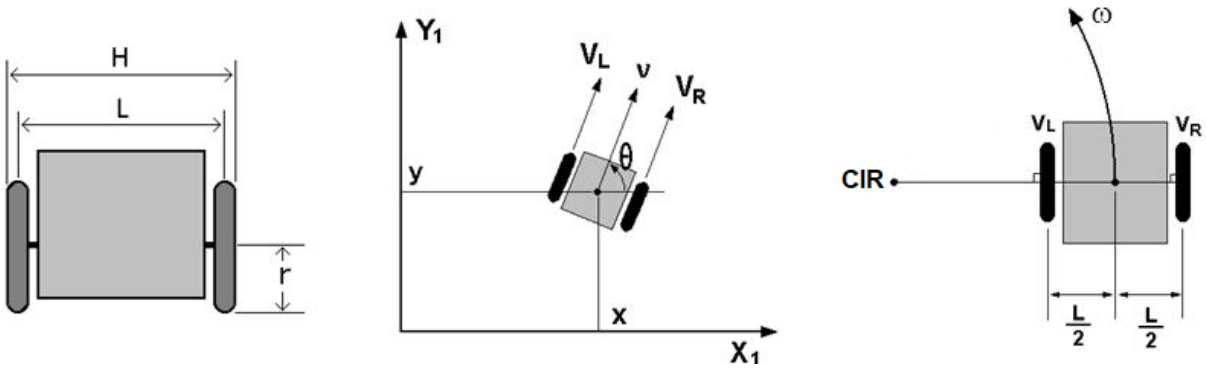


Figura 2.10: Modelo geométrico do robô diferencial. Fonte: Bianchi (2013).

Sendo assim, o modelo diferencial apresenta a seguinte relação:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} r\omega_R \\ r\omega_L \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} V_R \\ V_L \end{bmatrix}, \tag{2.5}$$

onde r é o raio da roda, L é distância entre as rodas, ω_R e ω_L são as velocidades angulares das rodas direita e esquerda, respectivamente. As velocidades lineares das rodas direita e esquerda são representadas pelas variáveis V_R e V_L, respectivamente.

Multiplicando as matrizes, tem-se que:

$$v(t) = \frac{V_R(t) + V_L(t)}{2}, \tag{2.6}$$

$$w(t) = \frac{V_R(t) - V_L(t)}{L}. \tag{2.7}$$

Relacionando as equação 2.4 e 2.5, tem-se que as velocidade lineares $\dot{x}$ e $\dot{y}$, e a velocidade angular do robô $\dot{\theta}$ podem ser expressas da seguinte forma:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} cos(\theta) & 0 \\ sen(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} V_R \\ V_L \end{bmatrix}.$$

Multiplicando as matrizes, tem-se que:

$$\dot{x} = \frac{(V_R + V_L)}{2} cos(\theta), \tag{2.8}$$

$$\dot{y} = \frac{(V_R + V_L)}{2} sen(\theta), \tag{2.9}$$

$$\dot{\theta} = \frac{(V_R - V_L)}{L}. \tag{2.10}$$

Adquirindo os valores de ω_R e ω_L por meio dos *encoders* acoplados aos motores e utilizando a integração numérica de primeira ordem pelo método Euler (SICILIANO *et al.*, 2010) nas equações 2.8, 2.9 e 2.10, é possível obter os valores de posição e orientação ao longo do tempo:

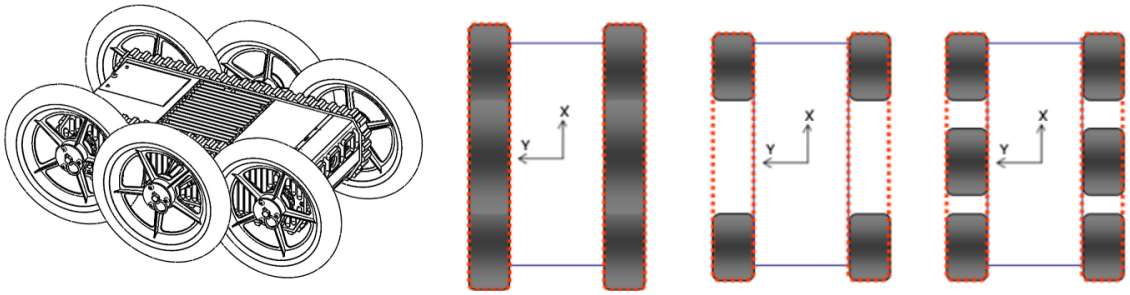
$$x(k+1) = v(k)cos(\theta)\Delta t + x(k), \tag{2.11}$$

$$y(k+1) = v(k)sin(\theta)\Delta t + y(k), \tag{2.12}$$

$$\theta(k+1) = w(k)\Delta t + \theta(k). \tag{2.13}$$

2.2.3 Modelo *skid-steer*

O modelo *skid-steer* representa veículos com pares de rodas ou esteiras, como ilustrado pela Figura 2.11. De maneira semelhante ao modelo diferencial, robôs do tipo *skid-steering* podem alcançar diferentes configurações por meio da movimentação das rodas. Velocidades idênticas em ambos os lados proporcionam uma translação linear e velocidades distintas fazem com que o robô realize curvas. Para girar em torno do seu próprio eixo, é necessário enviar velocidades de sinais inversos nos lados direito e esquerdo do veículo.



(a) (b)

Figura 2.11: (a) EspeleoRobô; (b) modelo *skid-steer* com esteira, 4 e 6 rodas, respectivamente. Fonte: Rocha (2018).

Com base nos trabalhos apresentados em (MANDOW *et al.*, 2007; MARTÍNEZ *et al.*, 2005), a cinemática de um robô *skid-steer* pode ser descrita da seguinte maneira:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = R_z(\theta) A \begin{bmatrix} V_R \\ V_L \end{bmatrix},$$

onde $R_z(\theta)$ é matriz de rotação elementar no eixo z pertencente ao grupo ortogonal $SO(3)$, A é a matriz de modelagem cinemática do dispositivo *skid-steer* que pertence ao grupo $\mathbb{R}^{3x2}$, e V_R e V_L representam as velocidades lineares nos eixos direito e esquerdo, respectivamente. As matrizes $R_z(\theta)$ e A são representadas da seguinte maneira:

$$R_z(\theta) = \begin{bmatrix} cos(\theta) & -sen(\theta) & 0 \\ sen(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$A = \frac{\alpha}{2x_{CIR}} \begin{bmatrix} 0 & 0 \\ x_{CIR} & x_{CIR} \\ -1 & 1 \end{bmatrix}.$$

Uma solução para encontrar a matriz A é descrita em (MANDOW *et al.*, 2007). Os autores propõem experimentos utilizando os centros instantâneos de rotação, ilustrados

pela Figura 2.12

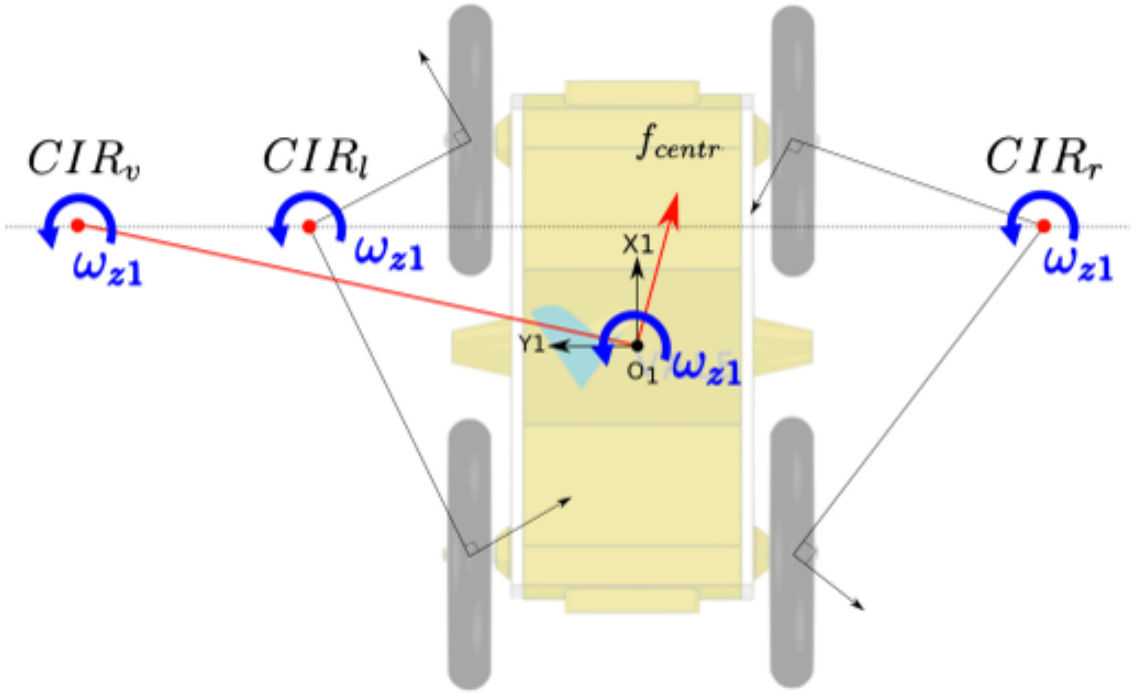


Figura 2.12: Localização dos CIRs dos eixos do veículos. Fonte: Rocha (2018).

Para ajustar o modelo cinemático do dispositivo, os autores encontram valores de forma empírica para o fator de correção α e x_{CIR}, calculados das seguintes maneiras:

$$\alpha = \frac{2d}{\int V_R(t)dt + \int V_L(t)dt}, \tag{2.14}$$

$$x_{CIR} = \frac{\int V_R(t)dt - \int V_L(t)dt}{2\theta^*}, \tag{2.15}$$

onde d é a distância percorrida pelo robô movimentando em linha reta, θ^* é o ângulo de rotação em torno do próprio eixo; $V_R(t)dt$ e $V_L(t)dt$ são as velocidades instantâneas dos eixos direto e esquerdo, respectivamente, durante os movimentos. Dessa forma, dois tipos de experimentos são realizados para se determinar os parâmetros. Para encontrar o fator de correção α, são conduzidos testes de locomoção em linha reta, aplicando velocidade de módulos e sentidos iguais em todas as rodas do robô. Enquanto para x_{CIR} são executados testes de rotação pura (em torno do próprio eixo), aplicando velocidades de módulos iguais, por em sentidos diferentes (COTA, 2019).

2.2.4 Estabilidade

Nesta dissertação a estabilidade é tratada como a capacidade de um dispositivo robótico em se manter apto à se movimentar, sem tombar. Tombamento é uma situação indesejada,

onde um ou mais mecanismos de atuação perdem efetividade na locomoção e/ou o chassi do robô colide com o terreno (ROCHA, 2018). Esse tema é relevante pois o dispositivo utilizado nesse trabalho, EspeleoRobô, é constantemente requerido para inspecionar em ambientes restritos ao acesso humano e, caso haja o tombamento do mesmo, poderá não haver maneira de resgatá-lo.

É possível avaliar a estabilidade de um dispositivo por meio de diferentes métricas, classificadas em dinâmicas e estáticas. As métricas que avaliam a estabilidade de forma dinâmica consideram as forças atuantes no dispositivo como gravidade, força centrípeta, torque, entre outras. As que avaliam a estabilidade de maneira estática, por sua vez, consideram apenas a força da gravidade, e são válidas em dispositivos atuando em baixas velocidade e com ação de distúrbios irrelevantes. Como o EspeleoRobô é utilizado para inspecionar ambientes desconhecidos e não-estruturados, apenas velocidades inferiores à 1m/s são desenvolvidas. Sendo assim, nessa dissertação a análise de estabilidade é realizada de forma estática.

Existem diversas métricas na literatura que estudam a estabilidade de dispositivos robóticos, sendo que cada uma leva em consideração características como configuração geométrica, forças atuantes, altura do centro de gravidade, entre outras. No entanto, a métrica estudada nesse trabalho é a do ângulo de tombamento.

2.2.5 Ângulo de tombamento

O ângulo de tombamento (AT) é uma métrica que indica diretamente qual é o menor ângulo que o dispositivo pode ser rotacionado até capotar (PAPADOPOULOS e REY, 1996). Sendo assim, é calculado o vetor Θ_n que contém com os ângulos necessários para rotacionar o dispositivo até seu limiar de estabilidade sobre as n arestas de sua borda de suporte. O valor de AT é o menor valor de Θ_n.

Considerando uma ilustração bidimensional do EspeleoRobô sobre um plano inclinado (Figura 2.13), para cara ponto de contato F_n nesse exemplo é traçado um vetor $\vec{G_n}$ que o liga ao centro de gravidade (CG). Ainda, a partir de cada F_n são descritos os vetores $\vec{G'_n}$ que possuem módulos iguais a seus respectivos $\vec{G_n}$, porém apontam no mesmo sentido de $\vec{Z}$. O vetor $\vec{Z} = [0\ 0\ 1]$ é o vetor diretor do eixo Z do sistema de coordenadas atrelado ao sistema inercial considerado. O vetor Θ_n contém os ângulos entre os vetores $\vec{G_n}$ e $\vec{G'_n}$.

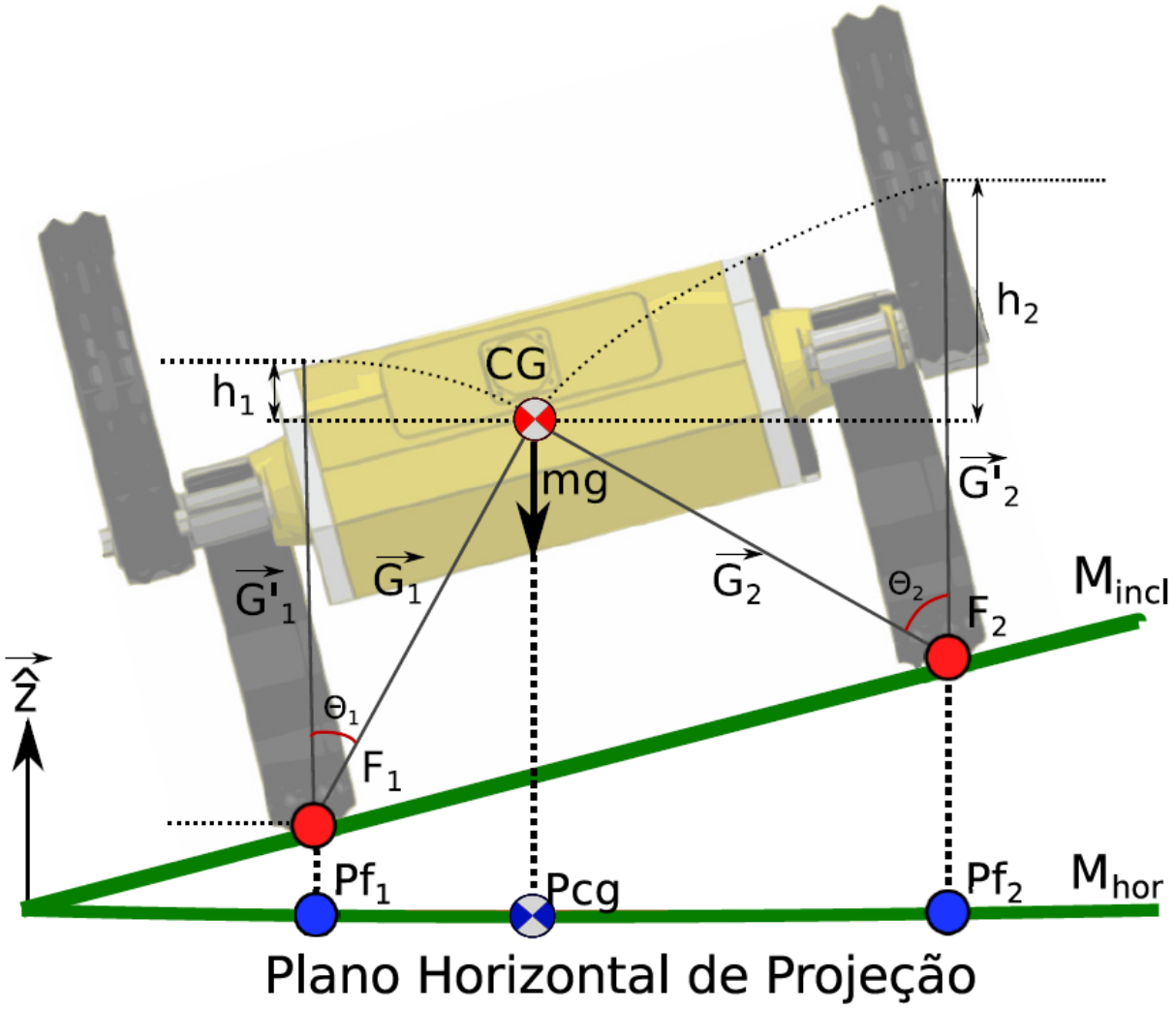


Figura 2.13: Representação bidimensional de dois pontos de apoio do EspeleoRobô e as variáveis de interesse relacionadas. Fonte: Rocha (2018).

Na ilustração apresentada é possível perceber que o EspeleoRobô está mais propenso a tombar em torno do ponto Pf_1 do que do ponto Pf_2. O dispositivo se torna instável quando o Pcg se encontra sobre ou fora do polígono suporte (PS), ilustrado pela Figura 2.14. Nesse exemplo, essa situação ocorre quando Pcg está sobre ou além de Pf_1 ou Pf_2. O fato de Θ_1 ser menor do que Θ_2 indica a maior facilidade do capotamento ocorrer sobre o ponto de apoio F_1, sentido plano abaixo.

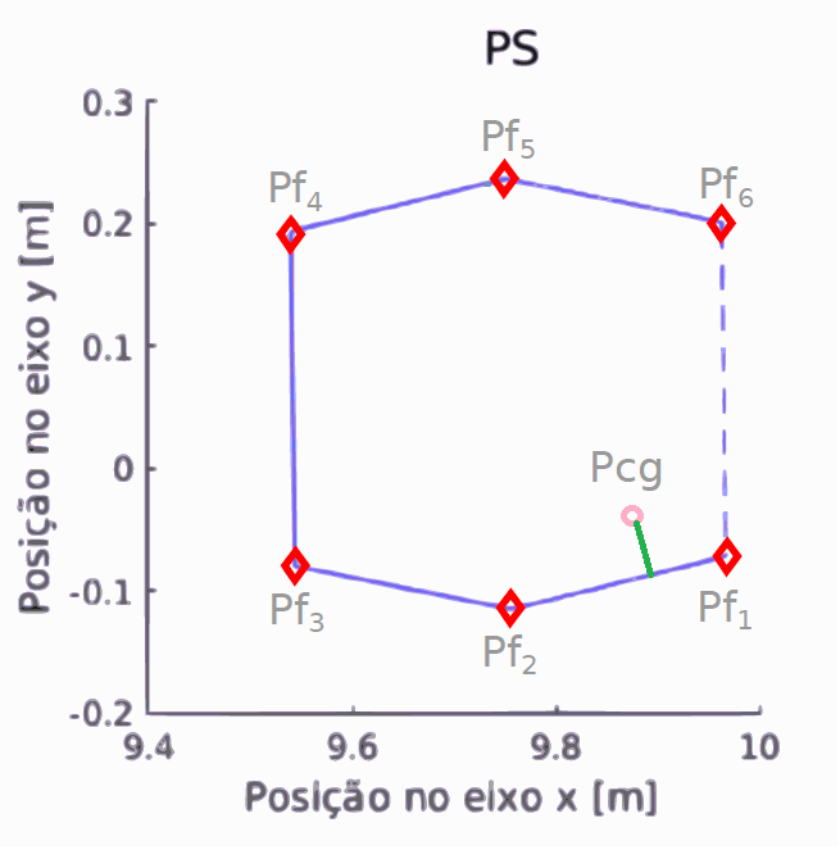


Figura 2.14: Representação do polígono suporte do EspeloRobô sobre um plano inclinado. Modificado de Rocha (2018).

Considerando agora o caso de CG e dois pontos genéricos F_1 e F_2 no espaço tridimensional (Figura 2.15), a reta $\overrightarrow{F_1F_2}$ que liga F_1 a F_2 é uma das arestas da borda de suporte. O *Plano1* pode ser descrito por $\overrightarrow{F_1F_2}$ e o vetor diretor do eixo Z ($\vec{Z}$). O vetor normal ao plano pode ser encontrado da seguinte maneira:

$$\vec{N}_{plano1} = \frac{\vec{Z} \times \overrightarrow{F_1F_2}}{\left|\vec{Z} \times \overrightarrow{F_1F_2}\right|}, \tag{2.16}$$

onde $\times$ é o operador que define o produto vetorial.

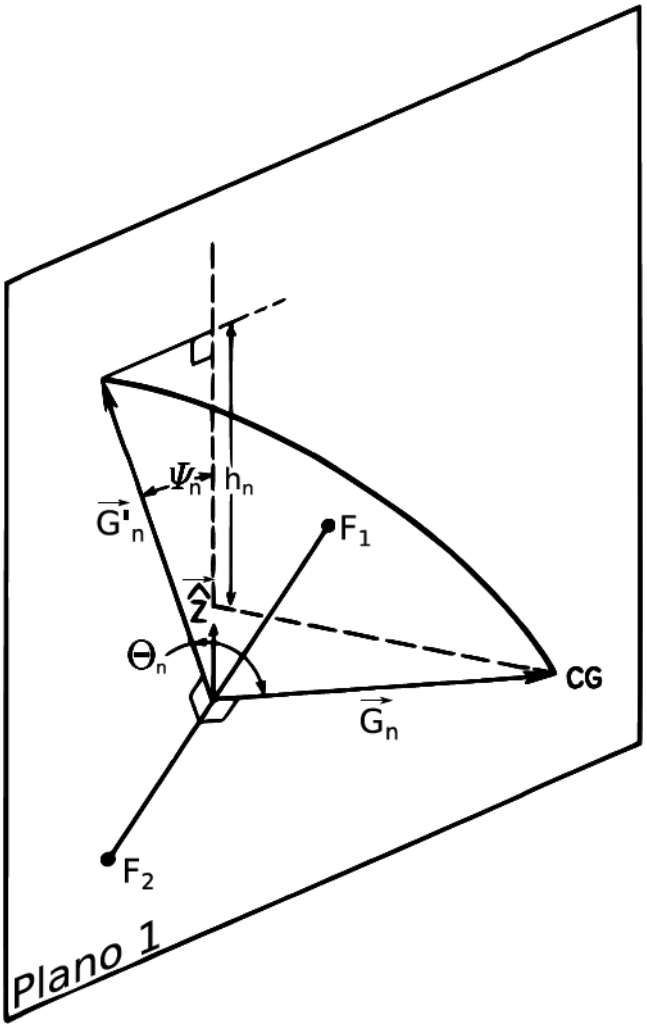


Figura 2.15: Representação tridimensional de dois pontos de apoio e as variáveis de interesse relacionadas. Fonte: Rocha (2018).

O vetor $\vec{G_n}$ descreve a menor distância entre $\overrightarrow{F_1F_2}$ e CG. É importante ressaltar que, na Figura 2.15, $\vec{G_n}$ está fora do *Plano1*, obtendo assim o vetor $\vec{G_n'}$. O ângulo Θ_n para essa aresta é dado pelo menor ângulo entre $\vec{G_n}$ e $\vec{G_n'}$, calculado da seguinte forma:

$$\Theta_n = arccos\left(\frac{\left|\vec{G_n}\cdot\vec{G_n'}\right|}{\left\|\vec{G_n}\right\|\left\|\vec{G_n'}\right\|}\right), \tag{2.17}$$

onde $\cdot$ é o operador que define o produto escalar.

O vetor $\vec{G_n'}$ pode ser encontrado da seguinte maneira:

$$\vec{G_n'} = \vec{G_n} - \left(\vec{G_n}\cdot\vec{N}_{plano}\right)\times\vec{N}_{plano}. \tag{2.18}$$

Uma vez calculado Θ_n para todas as arestas da borda de suporte, o valor do ângulo

de tombamento para a pose será igual ao menor valor encontrado:

$$AT = min\ \Theta_n,\ \forall\ n \in Q^*, \tag{2.19}$$

onde Q^* é conjunto de arestas do dispositivo.

Qualquer rotação maior ou igual a AT sobre a aresta de menor Θ_n da borda de suporte, fará com que o dispositivo tombe. Conhecendo os ângulos de tombamentos para diversas poses do robô no terreno, é possível prever uma eminente situação indesejada de capotamento.

2.3 Outros trabalhos sobre planejamento de caminhos

Muitos estudos têm sido realizados em relação ao planejamento de caminho para robôs móveis. Por exemplo, no trabalho realizado por Singh *et al.* (2000) é desenvolvido um sistema de navegação para um *rover* planetário utilizando visão binocular. Nesse trabalho, a navegação é dividida em dois níveis: global e local. A navegação global se encarrega de levar o veículo até o destino, enquanto a navegação local se preocupa com o desvio de obstáculos. Os algoritmos de navegação global e local utilizados são o D* e o Morphin (SIMMONS *et al.*, 1996), respectivamente. O algoritmo Morphin determina para cada trecho do terreno os ângulos rolagem (*roll*), arfagem (*pitch*), o índice de irregularidade e o grau de confiança das medidas. Ambos algoritmos fazem avaliação da transversalidade do terreno e votam por meio de pesos nas melhores ações para o robô. Para validação do sistema, foram realizados testes em campos com o *rover* Bullwinkle, onde o veículo dirigiu por 100m em uma velocidade de 15cm/s. Foram comparados vários testes utilizando diferentes pesos para os algoritmos de navegação global e local. Como resultado, quanto maior o valor do peso para o D*, mais perto o veículo chegou dos obstáculos. Isso geralmente resulta em um caminho mais curto, porém, a chance de colisão do veículo com os obstáculos é maior. No entanto, se o D* tem um peso de valor muito inferior ao Morphin, o robô às vezes não encontra o destino.

Em Ferguson *et al.* (2004) os autores desenvolveram um sistema robótico para realizar o mapeamento de minas abandonadas. O veículo chamado de Groundhog é equipado com computador embarcado, sensores *lasers*, de gás e equipamento de filmagem. Para navegação, o mapeamento 3-D obtido pelos *lasers* são convertidos em mapas de elevação utilizando representações de campos aleatórios de Markov (LI, 1994). Para encontrar os caminhos navegáveis os autores utilizaram o algoritmo A*. O sistema realiza escaneamentos consecutivos à medida que robô navega pela mina e os destinos são definidos pelas áreas mais distantes robô. Se o sistema não encontrar nenhum caminho navegável em um alcance de 2,5m, o robô decide dar meia volta e retornar ao ponto inicial exploração. O sistema robótico desenvolvido foi testado em três minas abandonadas, nas quais duas

eram inacessíveis para pessoas. Em um dos testes realizados com o Groundhog na mina de Mathies na Pensilvânia, a exploração obteve sucesso com o robô em modo autônomo. Nessa visita, o robô percorreu 308m da mina até encontrar um obstáculo no percurso que impediu sua passagem. Como esperado, Groundhog decidiu retornar ao ponto de origem, finalizando assim a exploração.

A superfície dos planetas conhecidos é composta por material escorregadio, onde o deslizamento das rodas do veículo podem fazer com que o mesmo fique preso. Como a dinâmica de escorregamento das rodas dependem de fatores como a velocidade do veículo, características do solo e da interação das rodas com o solo, incorporar essas questões no planejamento de caminho não é uma tarefa fácil. Sendo assim, no trabalho descrito em Ishigami *et al.* (2007) é apresentado um algoritmo de planejamento de caminho para um *rover* planetário que leva em consideração o escorregamento das rodas do veículo. Para encontrar os melhores trajetos os autores utilizaram o algoritmo de Dijkstra considerando três índices em sua função de custo: inclinação e irregularidade do terreno, e distância percorrida. Esses índices são normalizados e a relação entre eles é ajustada por meio de pesos. Dessa forma, o algoritmo de planejamento de caminho pode considerar a relevância de cada índice ao gerar soluções. Como resultado, foram avaliados dois trajetos obtidos pelo algoritmo onde manteve-se constante os pesos dos índices, porém, com restrições de inclinação de 15,0° e 7,5° em relação aos ângulos de rolagem (*roll*) e arfagem (*pitch*) do robô, respectivamente. Ambas soluções obtiveram resultados satisfatórios. No entanto, o caminho com menor valor de restrição de inclinação mostrou ser uma solução mais segura, apesar de ser um percurso mais longo. O trabalho apresentado pelos autores é similar ao trabalho descrito nesse documento. No entanto, nessa dissertação o ambiente é representado por meio de malhas triangulares ao invés de mapas de elevação. Ainda, o algoritmo proposto considera os pontos de contato do robô com o terreno para análise de estabilidade e o giro das rodas para o cálculo da métrica de consumo energético aos gerar caminhos soluções.

No estudo realizado por Raja *et al.* (2015) é proposto um algoritmo de planejamento de movimento para um *rover* com seis rodas em terrenos irregulares. Esse algoritmo utiliza o método de campos de potenciais para representar o ambiente, onde é apresentada uma função de gradiente que considera as forças de atração, repulsão, tangencial e gradiente. A força de gradiente varia de acordo com os ângulos rolagem, arfagem e guinada, que são obtidos por meio do modelo cinemático do veículo. O algoritmo considera essa força de modo a evitar que o *rover* navegue por locais onde o gradiente tenha valor elevado, resultando em um caminho mais seguro. São atribuídos pesos para os componentes da função de gradiente, que por sua vez são otimizados utilizando algoritmos genéticos. O algoritmo também avalia as velocidades das rodas do veículo para garantir a estabilidade e prevenir escorregamento. São comparados percursos que utilizam quatro diferentes métricas como função de custo do algoritmo: consumo de energia, escorregamento das rodas,

força de tração e distância percorrida. Por meio de simulações e testes experimentais, os autores concluíram que o método proposto é capaz de gerar melhores trajetos em terrenos irregulares se comparado com o método convencional de campos de potenciais.

Em Jeddisaravi *et al.* (2016) os autores propõem um planejamento de caminho multi-objetivo para robôs móveis baseado no algoritmo A*. O trabalho tem foco na aplicação de exploração planetária, na qual o ambiente possui fendas, colinas e composições rochosas. O objetivo do algoritmo proposto é encontrar caminhos de forma a minimizar métricas de dificuldade, risco e elevação entre a posição inicial até posição final. Os caminhos devem ser o mais curto possíveis para reduzirem o tempo de exploração e a energia consumida, e ao mesmo tempo seguro de modo a evitar problemas para o robô durante a navegação. Os autores também utilizaram pesos para ajustar a relação entre as métricas e confirmam a aplicabilidade do algoritmo por meio de simulações em dois cenários distintos.

Capítulo 3

Metodologias para planejamento de caminhos

Neste capítulo são apresentadas as metodologias utilizadas para o desenvolvimento do trabalho, que consiste na definição dos modelos de representação do ambiente e do robô, do algoritmo planejamento de caminho e das métricas utilizados nessa dissertação (Figura 3.1).

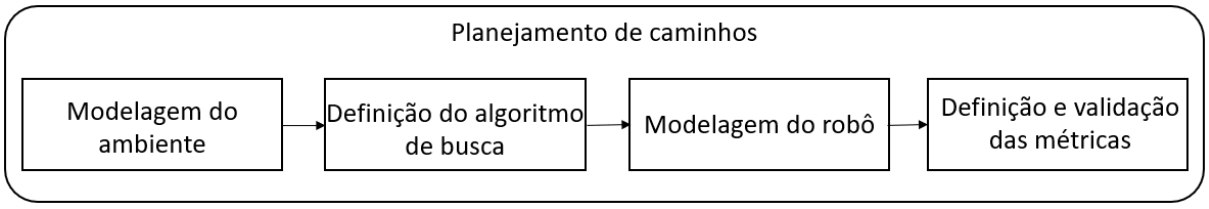


Figura 3.1: Representação das metodologias para planejamento de caminhos.

Nesta dissertação são adotadas as seguintes considerações:

- são utilizados mapas conhecidos com representação discreta com base na geometria do ambiente, onde não são tratadas incertezas de modelagem;
- toda área do terreno é trafegável;
- a pose do robô com respeito ao mapa é conhecida a todo momento;
- os coeficientes de atrito do terreno são constantes.

São avaliados percursos obtidos por meio da otimização das métricas de distância percorrida, transversalidade do terreno e consumo energético do robô utilizando o *software* Matlab. Ainda, é proposta uma função de custo que considera múltiplos objetivos durante o planejamento de caminho. A relação entre os objetivos é ajustada por meio de pesos definidos pelo operador. Assim, o algoritmo de planejamento de percurso avalia a

relevância de cada métrica ao encontrar trajetos ligando as posições inicial e final. Por fim, é realizado um estudo para analisar a interação do robô com o terreno utilizando os *softwares* Pybullet e V-REP. Assim, o algoritmo tenta prevenir o tombamento do robô durante a exploração e analisa se os caminhos são factíveis.

3.1 Modelo de representação do ambiente

O modelo de representação do ambiente adotado nesta dissertação é o de malhas triangulares, onde são utilizados mapas 3-D artificiais e um mapa de cavidade real. Os mapas artificiais foram gerados para validar o algoritmo de forma visual, o que poderia não ser intuitivo em uma mapa de terreno acidentado. O mapa de cavidade utilizado foi gerado a partir de uma nuvem de pontos coletada pela equipe de espeleologia da Vale em visitas em áreas com cavidades naturais. Esse mapa é relevante para o estudo pois a empresa recorrentemente precisa inspecionar áreas com características similares. Um exemplo dessa representação está ilustrado na Figura 3.2.

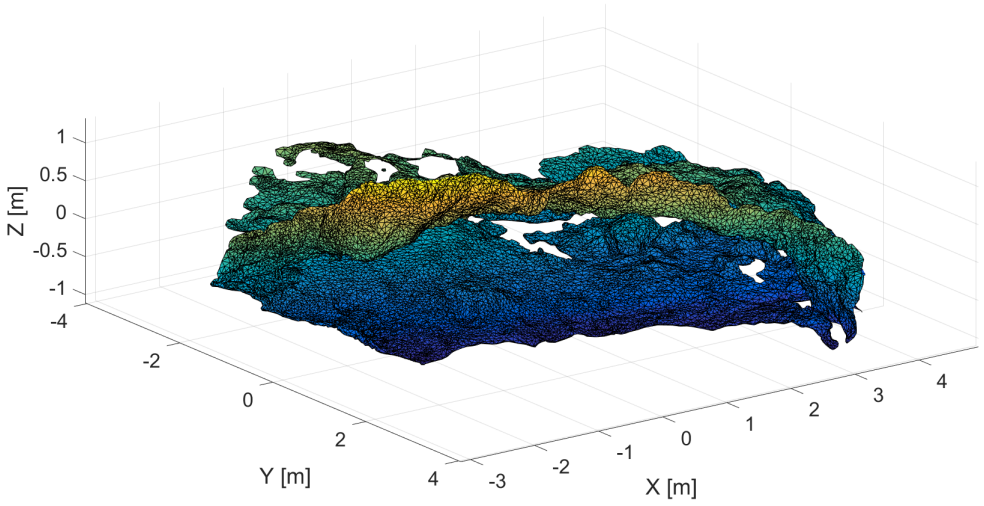


Figura 3.2: Mapa 3-D de uma cavidade representado por malhas triangulares. As cores das malhas estão relacionadas com o eixo Z.

3.2 Algoritmo de busca

De acordo com o modelo de representação do ambiente adotado nessa dissertação, o algoritmo de Dijkstra foi escolhido para o desenvolvimento do estudo por ser ótimo e adequado para problemas de busca em grafos com arcos de pesos fixos, distintos e não-negativos. Esse algorítimo tem convergência garantida para todos os casos, enquanto o A* só é ótimo se a heurística $h(n)$ for admissível. Com relação à métrica de distância

percorrida, uma heurística válida é a distância euclidiana entre as posições atual e final do robô. No entanto, não é trivial estimar as heurísticas para as métricas de transversalidade do terreno e consumo energético do robô, pois ambas são extremamente dependentes do perfil do terreno percorrido. Além disso, é possível utilizar o algoritmo de Dijkstra de maneira simples para otimizar múltiplas métricas simultaneamente.

Nessa dissertação é feita uma modificação no desempate do algoritmo original. Para a métrica de transversalidade do terreno, em caso de empate na escolha do menor custo, o algoritmo irá escolher o caminho mais curto e mais próximo da posição final. Essa modificação foi realizada para que os caminhos obtidos utilizando essa métrica em um mapa plano sem inclinação fossem os idênticos aos caminhos ótimos de distância percorrida e consumo energético do robô. Uma comparação resumida entre os principais algoritmos encontrados na literatura está apresentada na Tabela 3.1 e, o método escolhido, está descrito no Algoritmo 1.

Tabela 3.1: Tabela comparativa entre os principais algoritmos de planejamento de caminho encontrados na literatura.

Algoritmo	Característica principal	Custo dos arcos	Ótimo?
DFS	Adequado para problemas de pequena dimensão com múltiplas soluções	Igual	Não
BFS	Adequado para problemas de pequena dimensão com poucas soluções	Igual	Sim
RRT	Adequado para problemas com obstáculos no ambiente e restrições diferenciais	Igual	Não
A*	Adequado para problemas com heurística admissível	Fixo	Sim (caso a heurística seja admissível)
Dijkstra	**Adequado para problemas com grafos com arestas de peso não negativo**	**Fixo**	**Sim**
D*	Adequado para ambientes parcialmente conhecidos, estáticos ou dinâmicos	Variável	Sim (caso a heurística seja admissível)

Algoritmo 1: Pseudocódigo do Algoritmo de Dijkstra modificado.

1: Adicionar o nó inicial na lista de nós ABERTA e calcular a função de custo $f(n) = g(n)$.
2: Remover da lista ABERTA o nó com o menor valor de $f(n)$ e o colocar na lista de nós FECHADA. Esse é o nó de índice n. Em caso de empate, decidir pelo nó mais próximo do destino e que possui menor distância percorrida.
3: **se** $n = destino$ **então**
4: Usar os ponteiros dos nós para obter o caminho solução.
5: FIM
6: **senão**
7: Determinar todos os nós sucessores (filhos) de n e calcular $f(n)$ para cada sucessor que não esteja na lista FECHADA.
8: Associar os valores de $f(n)$ calculados aos nós sucessores que não estejam na lista de nós ABERTA ou FECHADA, colocá-los na lista ABERTA e adicionar ponteiros apontados ao nó pai de índice n.
9: Associar o menor valor de $f(n)$ a cada nó sucessor já contido na lista de nós ABERTA (mínimo(novo $f(n)$, velho $f(n)$)). Modificação: em caso de empate para a métrica de transversalidade do terreno, o algoritmo irá escolher o caminho mais curto e mais próximo da posição final.
10: Retornar ao passo 2.
11: **fim se**

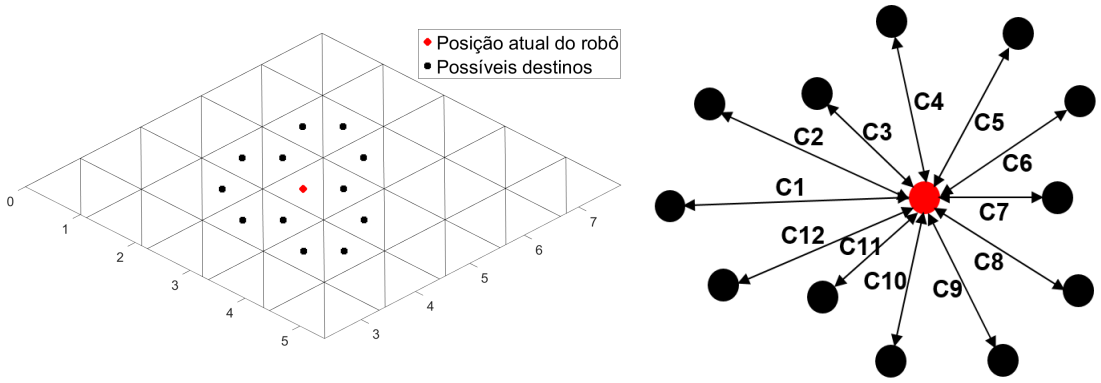
3.3 Representação do robô

Por motivos de simplificação, inicialmente o robô é considerado como uma partícula para o planejamento de caminho. No entanto, posteriormente o algoritmo irá considerar o arquitetura do robô para encontrar caminhos mais compatíveis e prevenir tombamentos.

3.3.1 Robô representado como uma partícula

Nesta etapa da dissertação não é considerado o modelo cinemático do EspeloRobô. No entanto, sabe-se que o robô possui seis rodas e é um veículo do tipo *skid-steering*. Ele é capaz de locomover para frente e para trás, e fazer rotações no seu próprio eixo. Sendo assim, o EspeleoRobô, teoricamente, é capaz de seguir qualquer caminho livre de obstáculos no solo por meio de uma sequência de movimentos lineares e angulares. Como proposto em Ge e Cui (2000), nessa etapa o robô é considerado como uma partícula para o planejamento de caminho. O robô pode se deslocar do centro de uma malha até o centro outra com vértice em comum. As malhas são definidas como nós, e os arcos ligando os nós adjacentes são associados aos custos de acordo com a métrica utilizada, exceto para a

métrica de transversalidade do terreno, onde cada nó possui um custo independente. As malhas nas quais o robô pode se deslocar a partir um ponto de referência estão ilustradas na Figura 3.3.



(a) (b)

Figura 3.3: (a) Representação da posição atual do robô e dos possíveis destinos; (b) grafo correspondente com os custos de acordo com a métrica utilizada.

3.3.1.1 Métrica de distância percorrida considerando o robô como uma partícula

Esta métrica tem como objetivo encontrar o caminho mais curto entre as posições inicial e final definidas para o robô em um terreno acidentado. A métrica de distância percorrida é calculada da seguinte maneira:

$$D_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}, \tag{3.1}$$

onde D_{ij} é a distância euclidiana entre os nós adjacentes de índices i e j. A função de custo utilizando essa métrica é definida pela seguinte equação:

$$C_1(p) = \sum_{n=p} D_{ij}(n), \tag{3.2}$$

onde o percurso p consiste em uma série de nós vizinhos dado por $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. O trajeto mais curto p_{dist} será encontrado de forma a satisfazer a seguinte equação de minimização do algoritmo de Dijkstra:

$$min\ C_1(p) = C_1(p_{dist}). \tag{3.3}$$

3.3.1.2 Métrica de transversalidade do terreno considerando o robô como uma partícula

A métrica de transversalidade visa obter o caminho mais plano entre as posições inicial e final definidas pelo usuário. Essa métrica pode ser calculada da seguinte forma:

$$T_i = arccos\left(\frac{\left|\vec{N}_i \cdot \vec{Z}\right|}{\left\|\vec{N}_i\right\|\left\|\vec{Z}\right\|}\right), \tag{3.4}$$

onde T_i é o ângulo agudo positivo entre a normal ($\vec{N}_i$) da malha triangular de índice i e o vetor diretor do eixo Z ($\vec{Z}$), ilustrado pela Figura 3.4.

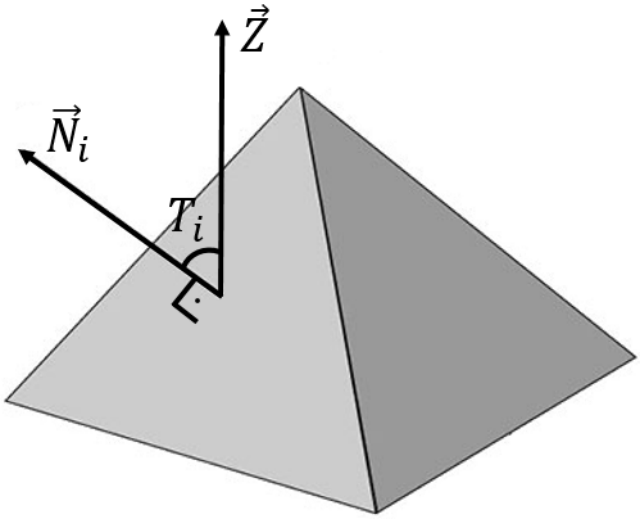


Figura 3.4: Ângulo agudo positivo entre a normal ($\vec{N}_i$) da malha triangular de índice i e o eixo Z.

A função de custo utilizando essa métrica é definida pela seguinte equação:

$$C_2(p) = \sum_{n=p} T_i(n), \tag{3.5}$$

onde o percurso p consiste em uma série de nós vizinhos dado por $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. O trajeto mais plano p_{tran}, cuja somatória dos ângulos é a menor possível, será encontrado de forma a satisfazer a seguinte equação de minimização do algoritmo de Dijkstra:

$$min\ C_2(p) = C_2(p_{tran}). \tag{3.6}$$

3.3.1.3 Métrica de consumo energético do robô considerando o robô como uma partícula

Esta métrica tem como objetivo descobrir o caminho com menor consumo energético entre as posições inicial e final definidas para o robô dentro da cavidade. Por motivos

de simplificação, está sendo considerado que o robô se move em movimento uniforme com velocidade constante. Isso significa que o robô gasta energia durante a frenagem para manter a uniformidade do movimento. A energia é calculada de maneira similar ao arrasto de um corpo em um plano inclinado. O diagrama de forças de um corpo em um plano inclinado está ilustrado pela Figura 3.5.

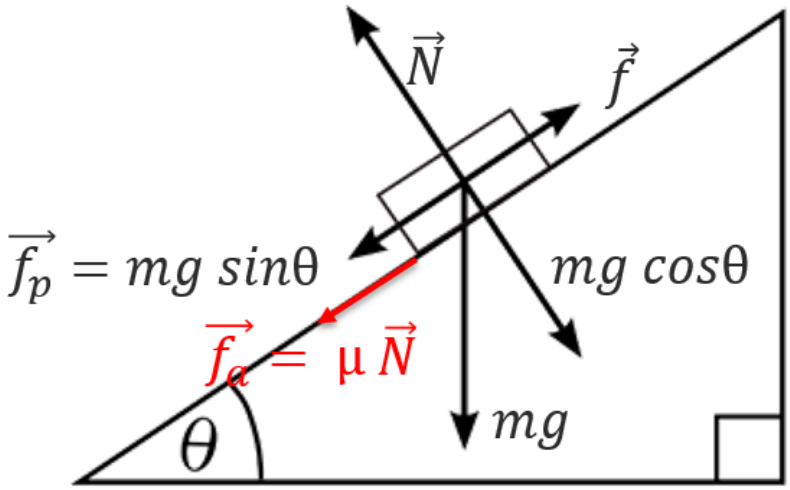


Figura 3.5: Diagrama de forças de um corpo em um plano inclinado.

O trabalho necessário para arrastar um corpo de massa m por uma distância D sobre um plano com inclinação θ é calculado pela seguinte equação:

$$\tau = \vec{f}.D,$$

$$\tau = (\vec{f_p} + \vec{f_a}).D,$$

$$\tau = (\mu.m.g.cos\theta + m.g.sin\theta).D, \tag{3.7}$$

onde $\vec{f_p}$ é a componente do peso no plano e $\vec{f_a}$ é a força de atrito.

Dessa forma, a métrica de energia é calculada da seguinte maneira:

$$E_{ij} = |\mu.m.g.cos\theta_{ij} + m.g.sin\theta_{ij}|.D_{ij}, \tag{3.8}$$

onde E_{ij} é a energia gasta para deslocar o robô entre os nós adjacentes de índices i e j, μ é coeficiente de atrito do terreno, m é a massa do robô, g é a aceleração da gravidade e θ_{ij} é o ângulo formado entre o vetor que liga os centros das malhas i e j e o plano horizontal definidos pelos eixos X e Y do terreno. Para fins de simplificação, os valores utilizados são $\mu = 1$, $m = 20kg$ e $g = 9,8m/s^2$. A função de custo utilizando essa métrica é definida

pela seguinte equação:

$$C_3(p) = \sum_{n=p} E_{ij}(n), \tag{3.9}$$

onde o percurso p consiste em uma série de nós vizinhos dado por $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. O trajeto mais econômico p_{ener} é encontrado desta forma:

$$min\ C_3(p) = C_3(p_{ener}). \tag{3.10}$$

3.3.1.4 Métricas combinadas considerando o robô como uma partícula

As métricas mencionadas anteriores são combinadas de forma a obter um caminho que considere a relevância de cada uma. A função de custo para essa abordagem é dada pela seguinte equação:

$$C_4(p) = \sum_{n=p} (P_d Nd_{ij} D_{ij}(n) + P_t Nt_{ij} T_i(n) + P_e Ne_{ij} E_{ij}(n)), \tag{3.11}$$

onde P_d, P_t e P_e são os pesos que ditam as prioridades entre os índices de distância percorrida, transversalidade do terreno e consumo energético do robô, respectivamente. Esses pesos são configurados pelo usuário de acordo com as prioridades da exploração. Por exemplo, se a prioridade da missão for o tempo, o usuário deve ajustar P_d com um valor superior em relação aos demais pesos. Em ambientes com alto risco de capotamento do robô, o usuário deve dar preferência ao peso P_t. Já em situações onde o consumo energético é crítico, o peso P_e deve ter prioridade em relação aos demais. A somatória desses pesos tem valor unitário. Nd_{ij}, Nt_{ij} e Ne_{ij} são os coeficientes de normalização, que por sua vez são obtidos de acordo com a vizinhança dos nós adjacentes, onde todos os custos são divididos pelo maior custo local de acordo com cada métrica. O caminho de navegação p_{comb} resultante será encontrado desta maneira:

$$min\ C_4(p) = C_4(p_{comb}). \tag{3.12}$$

3.3.2 Robô representado por modelo cinemático

Nesta etapa é considerado o modelo cinemático do EspeloRobô. Como mencionado anteriormente, o robô é um veículo do tipo *skid-steering* e é utilizado com a configuração de seis rodas. Ele é capaz de locomover para frente e para trás, e fazer rotações no seu próprio eixo. Na Figura 3.6 estão descritas as dimensões e os parâmetros do robô.

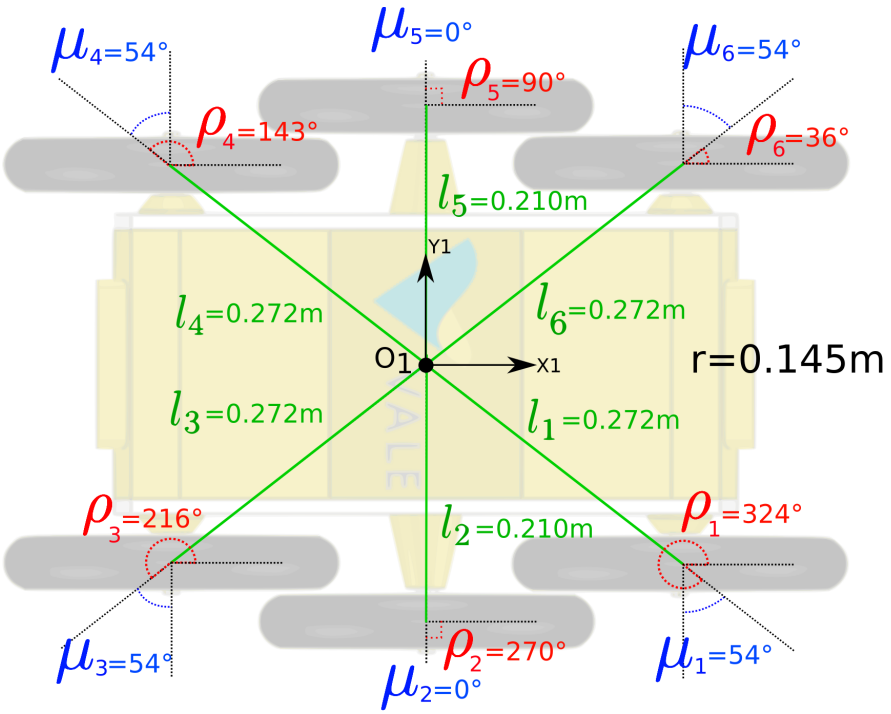


Figura 3.6: Representação das dimensões e parâmetros do EspeleoRobô.

3.3.2.1 Métrica de distância percorrida considerando o modelo cinemático do robô

A métrica de distância percorrida considerando a cinemática do robô incorpora o giro das rodas necessário para realizar deslocamentos lineares e angulares que o mesmo precisa fazer para se deslocar da posição inicial até a posição final. Assim, a métrica em questão é calculada da seguinte maneira:

$$Dc_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} + \frac{\beta_{ij}.2.\pi.l_5}{360} \mid 0 \leq \beta_{ij} \leq 180, \tag{3.13}$$

onde o primeiro termo é a distância euclidiana entre os nós adjacentes de índices i e j, β_{ij} é a rotação em graus que o robô precisa fazer entre os nós adjacentes de índices i e j, e l_5 é a metade da distância entre as duas rodas centrais do EspeleoRobô. A função de custo utilizando essa métrica é definida pela seguinte equação:

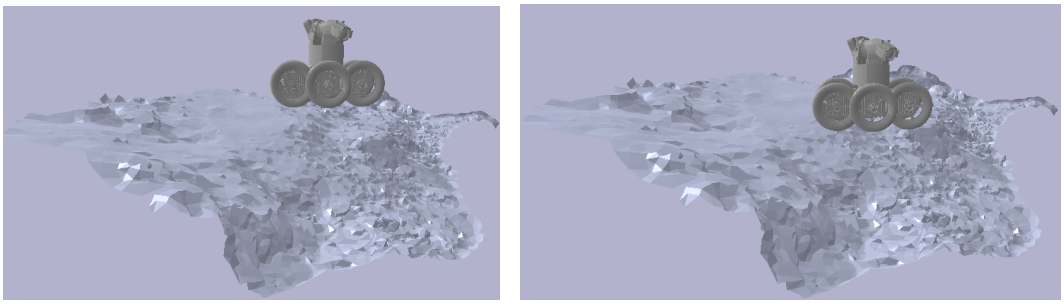
$$C_5(p) = \sum_{n=p} Dc_{ij}(n), \tag{3.14}$$

onde o percurso p consiste em uma série de nós vizinhos dado por $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. O trajeto mais curto pc_{dist} será encontrado de forma a satisfazer a seguinte equação:

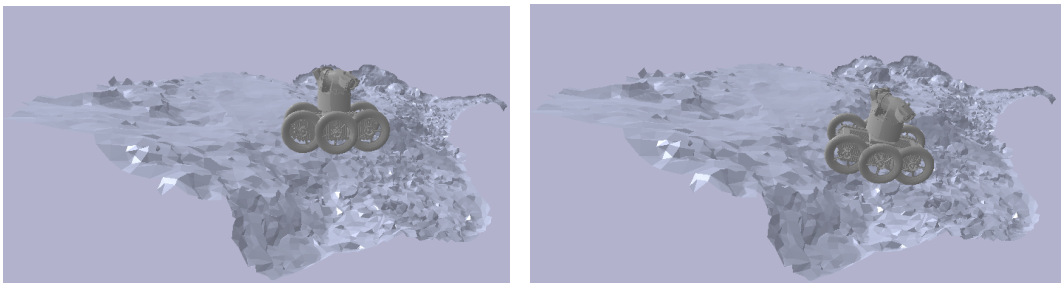
$$min\ C_5(p) = C_5(p_{dist}). \tag{3.15}$$

3.3.2.2 Métrica de transversalidade do terreno considerando o modelo cinemático do robô

A métrica de transversalidade considerando a cinemática do robô visa obter o caminho estável ou com menor risco de capotamento para o robô da posição inicial até a posição final, ambas definidas pelo usuário. Essa métrica é baseada no ângulo de tombamento do robô, que indica qual é o menor ângulo que o dispositivo pode ser rotacionado até capotar. Os ângulos de tombamento são calculados a partir do estudo da interação do robô com o terreno utilizando o simulador de física Pybullet. O robô é solto de uma determinada altura e assim que o mesmo se acomoda no terreno, o simulador retorna as posições dos pontos de contato das rodas do robô com o terreno e centro de gravidade do robô (Figura 3.7). Em caso de tombamento do robô durante a simulação, o simulador termina a simulação e retorna uma mensagem de erro. Esse *software* foi escolhido por ser uma ferramenta leve e capaz de gerar resultados com muita rapidez. É feita uma interação entre o Matlab e Pyblullet de modo que o Matlab execute automaticamente o código escrito que Python enquanto o algoritmo de planejamento de caminho encontra a solução.



(a) (b)



(c) (d)

Figura 3.7: Estudo da interação do robô com o terro utilizando o simulador Pybullet: posição inicial (a), posição dois (b), posição três (c) e posição final (d).

Uma vez obtidos os ângulos de tombamento do robô, essa métrica de transversalidade é calculada da seguinte forma:

$$Tc_{ij} = AT_{max} - min\ [AT_{ij}(n)] \mid n = 1, 2, ..., 6\ , \tag{3.16}$$

onde AT_{max} é o maior ângulo de tombamento possível, isto é, quando o robô está em um plano sem inclinação, e $AT_{ij}(n)$ é um vetor com os seis ângulos de tombamentos do robô quando o centro de gravidade (CG) do mesmo está posicionado sobre a malha triangular de índice i com orientação em direção à malha de índice j. A função de custo utilizando essa métrica é definida pela seguinte equação:

$$C_6(p) = \sum_{n=p} Tc_{ij}(n), \tag{3.17}$$

onde o percurso p consiste em uma série de nós vizinhos dado por $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. O trajeto com menor risco de capotamento pc_{tran} será encontrado de forma a satisfazer a seguinte equação:

$$min\ C_6(p) = C_6(pc_{tran}). \tag{3.18}$$

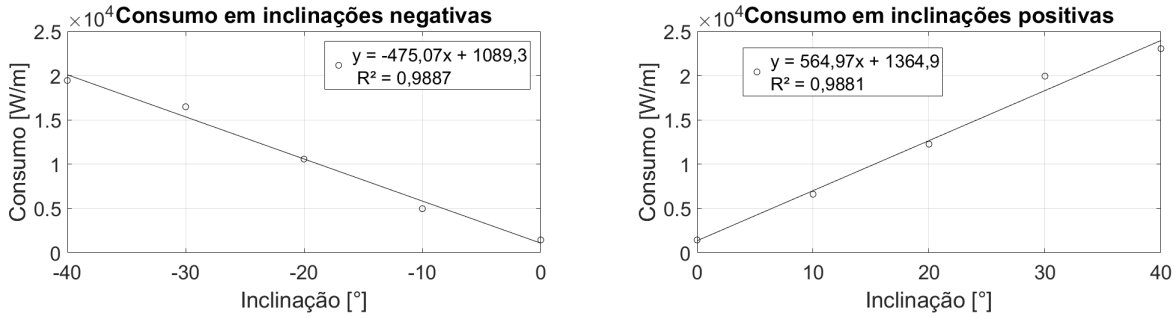
3.3.2.3 Métrica de consumo energético do robô considerando o modelo cinemático do robô

A métrica de consumo energético considerando a cinemática do robô leva em consideração as energias que o robô gasta ao se locomover em linha reta e fazer rotações no próprio eixo. Para tal, foram realizadas simulações utilizando o simulador V-REP. Nessas simulações o robô seguiu em linha reta por 50m em planos com diferentes inclinações com velocidade linear de 1m/s. Segundo Cherif (1999), os coeficientes de atrito dinâmico e estático para regiões não-escorregadias variam nos intervalos [0,2 - 0,6] e [0,4 - 0,9], respectivamente. Sendo assim, nesta dissertação os valores máximos, 0,6 e 0,9, são adotados para os atritos dinâmico e estático, respectivamente. Também foi realizado um ensaio para estimar o consumo médio do robô ao realizar rotações no próprio eixo. Nesse experimento o robô rotacionou por 20 vezes em torno do seu próprio eixo em um plano sem inclinação e o consumo médio para rotacionar 360° foi de 37.735,9W. Para cada cenário foram realizadas três simulações e o consumo final é a média aritmética das medidas.

Tabela 3.2: Consumos energéticos do robô obtidos nas simulações em planos inclinados.

Inclinação (°)	Consumo 1 (W/m)	Consumo 2 (W/m)	Consumo 3 (W/m)	Consumo médio (W/m)
-40	19323,54	19601,23	19449,98	19458,25
-30	16191,21	16322,80	16927,91	16480,64
-20	10689,28	10831,48	10214,55	10578,47
-10	4880,49	5081,28	4984,41	4982,06
0	1523,11	1405,62	1433,35	1454,07
10	6486,35	6745,29	6619,85	6617,16
20	12168,32	12247,14	12392,65	12269,37
30	20343,57	19599,94	19885,82	19943,11
40	22621,84	23218,00	23278,71	23039,52

Para encontrar o consumo energético do robô em outras inclinações foram realizadas regressões lineares, ilustradas na Figura 3.8.



(a) (b)

Figura 3.8: (a) Consumo energético do robô em planos com inclinações negativas; (b) consumo energético do robô em planos com inclinações positivas.

Dessa forma, a métrica de energia é calculada das seguintes maneiras:

$$Ec_{ij} = \left(\frac{37735,9\beta_{ij}}{360}\right) + (-475,07\beta_{ij} + 1089,3)D_{ij}, \forall\ \theta < 0, \tag{3.19}$$

$$Ec_{ij} = \left(\frac{37735,9\beta_{ij}}{360}\right) + (564,97\beta_{ij} + 1364,9)D_{ij}, \forall\ \theta \geq 0, \tag{3.20}$$

onde Ec_{ij} é a energia gasta para rotacionar e deslocar o robô entre os nós adjacentes de índices i e j, β_{ij} é a rotação em graus que o robô precisa fazer entre os nós adjacentes de índices i e j, θ_{ij} é o ângulo formado entre o vetor que liga os centros das malhas i e j e o plano XY, e D_{ij} é a distância euclidiana entre os nós de índices i e j. A função de custo utilizando essa métrica é definida pela seguinte equação:

$$C_7(p) = \sum_{n=p} Ec_{ij}(n), \tag{3.21}$$

onde o percurso p consiste em uma série de nós vizinhos dado por $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. O trajeto mais econômico pc_{ener} será encontrado desta forma:

$$min\ C_7(p) = C_7(pc_{ener}). \tag{3.22}$$

3.3.2.4 Métricas combinadas considerando o modelo cinemático do robô

As métricas considerando a cinemática do robô são combinadas de forma a obter um caminho que considere a relevância de cada uma. A função de custo para essa abordagem

é dada pela seguinte equação:

$$C_8(p) = \sum_{n=p}(P_d Nd_{ij} Dc_{ij}(n) + P_t Nt_{ij} Tc_{ij}(n) + P_e Ne_{ij} Ec_{ij}(n)), \tag{3.23}$$

onde P_d, P_t e P_e são os pesos que ditam as prioridades entre os índices de distância percorrida, transversalidade do terreno e consumo energético do robô, respectivamente. A somatória desses pesos tem valor unitário, resultando em um total de 100%. Nd_{ij}, Nt_{ij} e Ne_{ij} são os coeficientes de normalização, que por sua vez são obtidos de acordo com vizinhança dos nós adjacentes, onde todos os custos são divididos pelo maior custo local de acordo com cada métrica. O caminho de navegação pc_{comb} resultante será encontrado desta maneira:

$$min\ C_8(p) = C_8(pc_{comb}). \tag{3.24}$$

Capítulo 4

Resultados e discussões

Neste capítulo são apresentados os resultados obtidos utilizando as métricas descritas no Capítulo 3, seguido por discussões.

4.1 Resultados para o planejamento de caminhos representando o robô como uma partícula

Nesta seção são apresentados os resultados do algoritmo de planejamento de caminhos utilizando as métricas que consideram o robô como uma partícula.

4.1.1 Validação das métricas no Matlab utilizando mapas artificiais e considerando o robô como uma partícula

Inicialmente, as métricas propostas considerando o robô como uma partícula foram validadas no Matlab utilizando dois mapas artificiais: um mapa plano sem ondulações e um mapa 3-D representados por malhas triangulares (Figuras 4.1 e 4.2). O mapa 3-D foi gerado a partir de curvaturas em forma de gaussianas para simular irregularidades no terreno. Esses mapas foram elaborados dessa forma para validar o algoritmo de forma visual, o que poderia não ser intuitivo em uma mapa de terreno acidentado. Os pesos utilizados para os caminhos obtidos com métricas combinadas foram escolhidos de forma empírica. Os caminhos ótimos obtidos para cada métrica no mapa plano e no mapa 3-D estão representados nas Figuras 4.1 e 4.2, respectivamente. Os valores de distância percorrida, ângulo máximo, energia consumida e custo total normalizado de cada caminho estão apresentados nas Tabelas 4.1 e 4.2. O custo total normalizado corresponde a soma dos custos normalizados das três métricas. Por se tratar de uma mapa plano, esse custo para os caminhos obtidos possui o mesmo valor independentemente dos pesos utilizados.

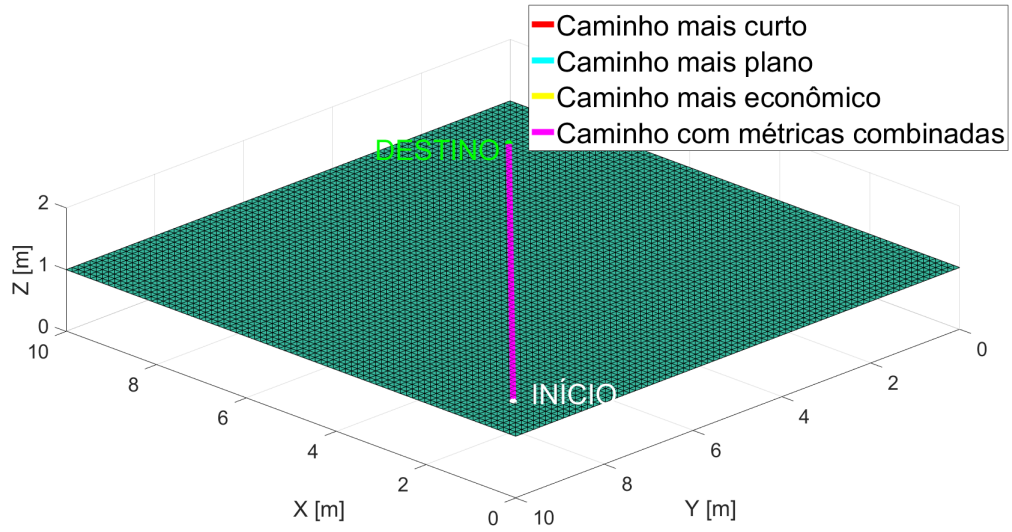


Figura 4.1: Caminhos ótimos obtidos utilizando o algoritmo de Dijkstra considerando as métricas de distância percorrida, transversalidade do terreno e consumo energético do robô em um mapa plano.

Tabela 4.1: Valores calculados para os caminhos ótimos obtidos nas simulações realizadas em um mapa plano sem ondulações.

Caminho	Distância percorrida (m)	Angulo máximo (°)	Energia consumida (J)	Custo total normalizado
Mais curto	10,893	0	2135,1	27,04
Mais plano	10,893	0	2135,1	27,04
Mais econômico	10,893	0	2135,1	27,04
Métricas combinadas	10,893	0	2135,1	27,04

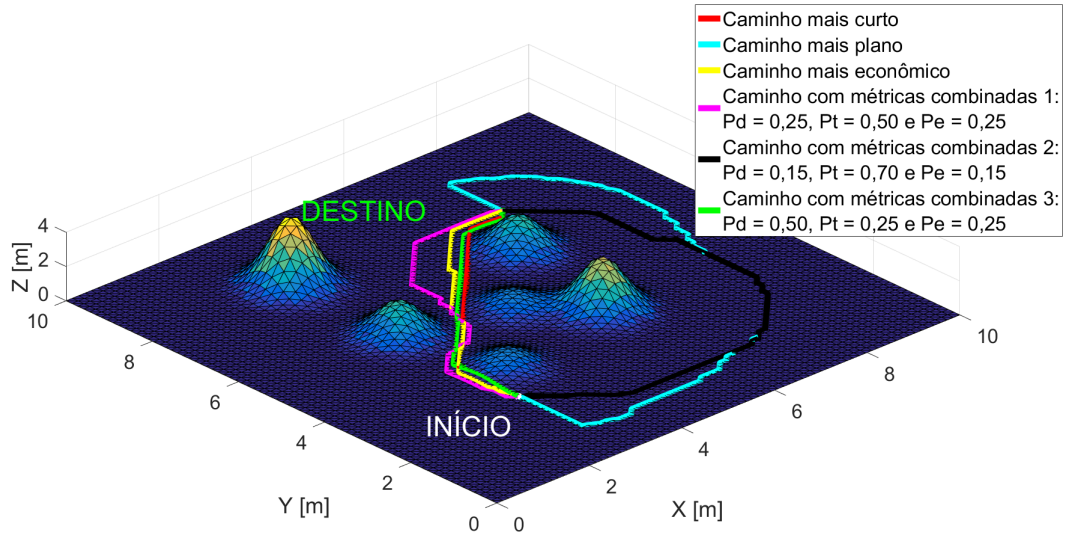


Figura 4.2: Caminhos ótimos obtidos utilizando o algoritmo de Dijkstra considerando as métricas de distância percorrida, transversalidade do terreno e consumo energético do robô em um mapa formado por curvaturas gaussianas.

Tabela 4.2: Valores calculados para os caminhos ótimos obtidos nas simulações realizadas em um mapa formado por curvaturas gaussianas.

Caminho	Distância percorrida (m)	Ângulo máximo (°)	Energia consumida (J)	Custo total normalizado 1	Custo total normalizado 2	Custo total normalizado 3
Mais curto	**7,43**	56,58	1671,20	34,99	37,47	32,32
Mais plano	16,47	**7,42**	3233,20	45,64	31,19	63,70
Mais econômico	7,61	39,65	**1585,70**	32,40	32,58	32,81
Métricas combinadas 1: Pd=0,25, Pt=0,50 e Pe=0,25	8,45	25,77	1725,10	**30,53**	27,06	35,26
Métricas combinadas 2: Pd=0,15, Pt=0,70 e Pe=0,15	12,04	7,42	2364,70	34,16	**23,97**	46,92
Métricas combinadas 3: Pd=0,50, Pt=0,25 e Pe=0,25	7,46	50,69	1652,30	34,12	36,26	**31,92**

4.1.2 Simulações no Matlab utilizando um mapa de cavidade natural e considerando o robô como uma partícula

Após concluída a etapa de validação, foram realizadas simulações em um mapa de cavidade natural. Os pesos utilizados para os caminhos com métricas combinadas foram os mesmos da etapa de validação. Os caminhos ótimos para cada métrica estão ilustrados na Figura 4.3. Os valores de distância percorrida, ângulo máximo, energia consumida e custo total normalizado de cada caminho estão apresentados na Tabela 4.3.

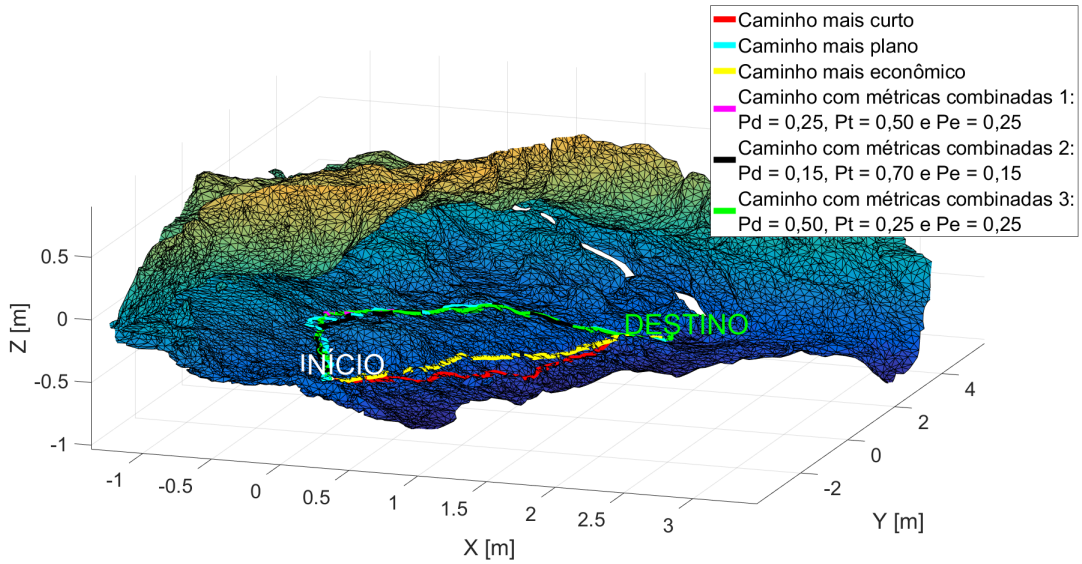


Figura 4.3: Caminhos ótimos obtidos utilizando o algoritmo de Dijkstra considerando as métricas de distância percorrida, transversalidade do terreno e consumo energético do robô em um mapa de cavidade natural. As cores das malhas estão relacionadas com o eixo Z.

Tabela 4.3: Valores calculados para os caminhos ótimos obtidos nas simulações realizadas em um mapa de cavidade natural.

Caminho	Distância percorrida (m)	Ângulo máximo (°)	Energia consumida (J)	Custo total normalizado 1	Custo total normalizado 2	Custo total normalizado 3
Mais curto	**4,94**	87,95	1080,10	50,20	50,19	61,87
Mais plano	6,14	**40,04**	1362,00	45,38	36,81	56,16
Mais econômico	4,99	88,85	**1058,30**	59,78	57,82	63,36
Métricas combinadas 1: Pd=0,25, Pt=0,50 e Pe=0,25	5,48	47,96	1222,40	**41,38**	34,63	49,92
Métricas combinadas 2: Pd=0,15, Pt=0,70 e Pe=0,15	5,53	47,96	1227,30	41,60	**34,37**	50,71
Métricas combinadas 3: Pd=0,50, Pt=0,25 e Pe=0,25	5,36	47,96	1192,30	42,19	36,52	**49,37**

4.1.3 Análise dos resultados considerando o robô como uma partícula

Como esperado, os caminhos ótimos obtidos na simulação em um mapa plano sem ondulações foram os mesmos para todas as métricas. Isso devido à modificação realizada no algoritmo de Dijkstra descrita na Seção 3.1. Ainda, analisando os resultados das Tabelas 4.2 e 4.3, é possível concluir que os caminhos obtidos pelo algoritmo de Dijkstra considerando as métricas propostas mostraram ser ótimos, visto que os melhores valores de distância percorrida, ângulo máximo e energia consumida estão relacionados aos trajetos que utilizam as respectivas métricas no planejamento de caminho. Os menores custos totais normalizados também estão relacionados com os caminhos que utilizam os respectivos pesos. É mais intuitivo chegar a essa conclusão de forma visual observando os resultados obtidos na etapa de validação do algoritmo, ilustrados na Figura 4.2. Ainda, é possível observar que os caminhos encontrados utilizando as métricas de forma combinada alcançaram valores intermediários de distância percorrida, ângulo máximo e energia consumida.

Analisando os resultados apresentados na Tabela 4.3, o caminho com métricas combinas 3 apresentou os melhores resultados, visto que esse caminho é apenas 8,50% mais longo que o percurso mais curto e é somente 12,66% menos econômico que o trajeto mais econômico. Isso se deve à tentativa do algoritmo em minimizar todos os atributos de forma simultânea, levando em consideração o peso atribuído a cada uma.

4.2 Resultados para o planejamento de caminhos considerando o modelo cinemático do robô

4.2.1 Validação das métricas no Matlab utilizando mapas artificias e considerando o modelo cinemático do robô

As métricas propostas considerando a cinemática do robô também foram validadas utilizando um mapa plano sem ondulações e um mapa 3-D representado por malhas triangulares e gerado a partir de curvaturas com forma de gaussianas (Figura 4.4). Os resultados da simulação utilizando o mapa plano foram idênticos aos da Figura 4.1. Foram utilizados diferentes pesos para obtenção dos caminhos com métricas combinadas. No entanto, o melhor resultado constatado foi utilizando $P_d = 0,25$, $P_t = 0,5$ e $P_e = 0,25$, visto que o caminho resultante apresentou os melhores valores para distância percorrida, ângulo mínimo de tombamento e consumo energético. Os caminhos ótimos obtidos para cada métrica estão representados na Figura 4.4. Os valores de distância percorrida, ângulo mínimo de tombamento, energia consumida e os custo total normalizado de cada caminho estão apresentados na Tabela 4.4. Para evitar que o robô navegue próximo das extremidades do mapa, são removidas do planejamento de caminhos as áreas próximas à borda num raio da largura do EspeleoRobô, ou seja, $2l_5$.

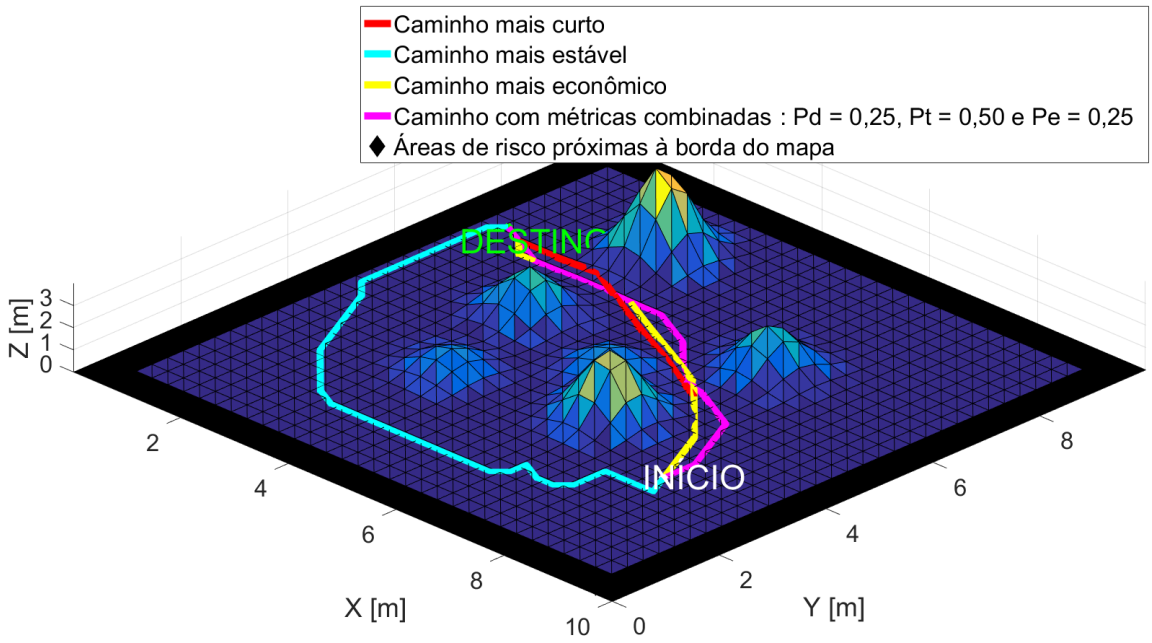


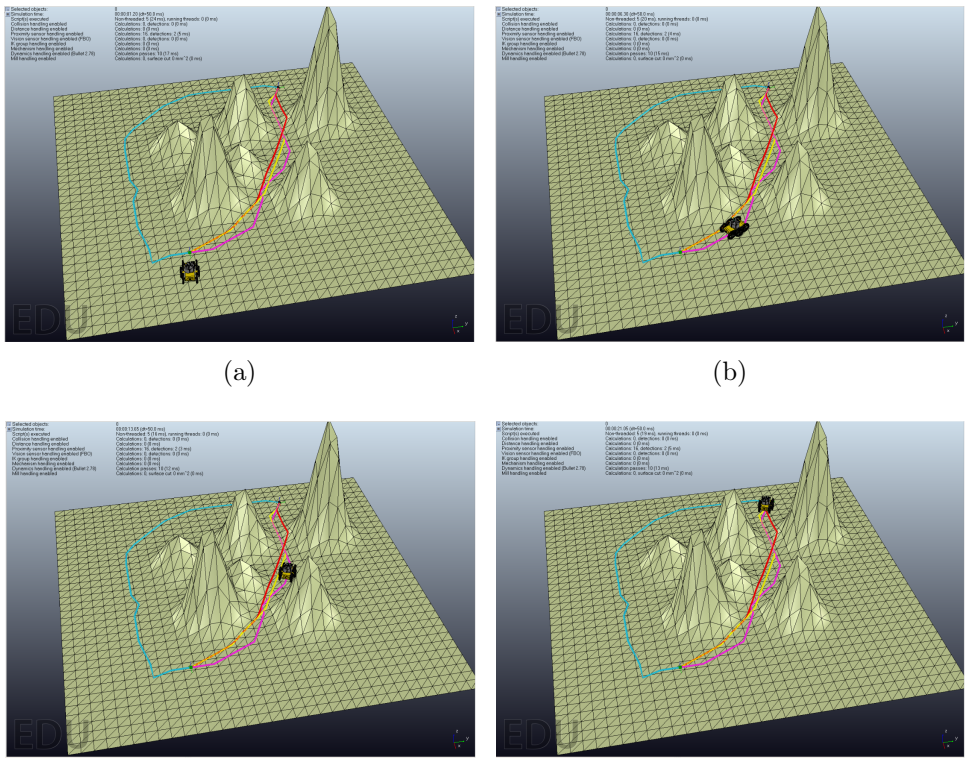
Figura 4.4: Caminhos ótimos obtidos utilizando o algoritmo de Dijkstra para as métricas de distância percorrida, transversalidade do terreno e consumo energético do robô em um mapa formado por curvaturas gaussianas considerando a cinemática do mesmo. As cores das malhas estão relacionadas com o eixo Z.

Tabela 4.4: Valores calculados para os caminhos ótimos obtidos nas simulações realizadas em um mapa formado por curvaturas gaussianas considerando a cinemática do robô.

Caminho	Distância percorrida (m)	Ângulo mínimo de tombamento (°)	Energia consumida (W)	Custo total normalizado
Mais curto	**9,21**	5,38	48504,00	5,89
Mais estável	15,72	**45,93**	96795,00	5,34
Mais econômico	9,42	18,96	**44316,00**	4,36
Métricas combinadas	9,96	39,95	49404,00	**3,52**

4.2.2 Simulações no V-REP utilizando um mapa 3-D artificial

Uma vez obtidos os caminhos ótimos para cada métrica, foram realizadas simulações no V-REP para garantir que os caminhos são factíveis pelo robô. É implementado um controle de navegação simples, onde as velocidades das rodas do robô variam de forma a manter uma velocidade linear constante, e a velocidade angular do robô varia de forma proporcional à rotação que o mesmo precisa fazer para seguir o caminho desejado. O valor desejado para velocidade linear foi definido como 0,5m/s. Como resultado, o robô conseguiu percorrer todos os caminhos encontrados pelo algoritmo de Dijkstra. No entanto, o robô apresentou dificuldade ao navegar pelos caminhos "Mais curto"e "Mais econômico". Isso porque esses caminhos passam por regiões com inclinações elevadas, dificultando o movimento do robô. Uma das simulações está ilustrada na Figura 4.5.



(a) (b)

(c) (d)

Figura 4.5: Seguimento de caminho em um mapa artificial utilizando o simulador V-REP: posição inicial (a), posição dois (b), posição três (c) e posição final (d).

4.2.3 Simulações no Matlab utilizando um mapa de cavidade natural e considerando o modelo cinemático do robô

Após concluída a etapa de validação, novamente foram realizadas simulações em um mapa de cavidade natural. Os pesos utilizados para obter o caminho com métricas combinadas foram mantidos (i.e., $P_d = 0,25$, $P_t = 0,5$, and $P_e = 0,25$). Os caminhos ótimos para cada métrica estão ilustrados na Figura 4.6. Os valores de distância percorrida, mínimo ângulo de tombamento, energia consumida e custo total normalizado de cada caminho estão apresentados na Tabela 4.5.

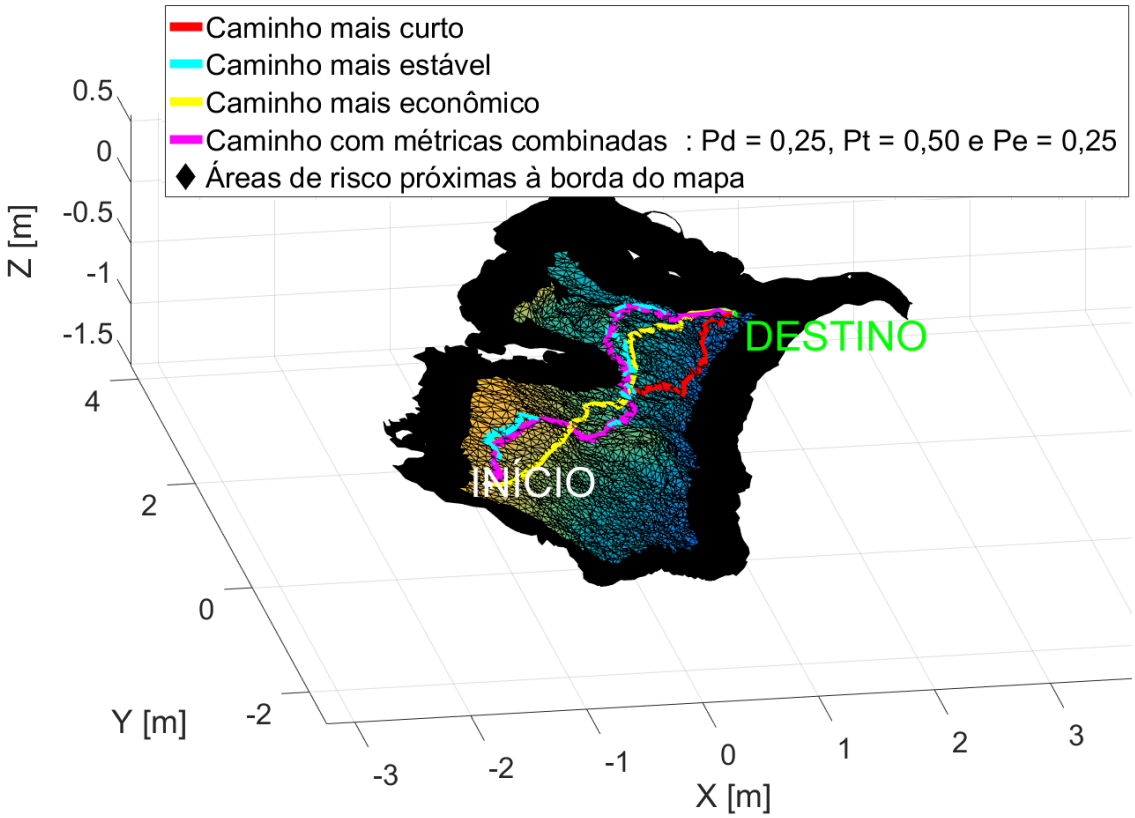


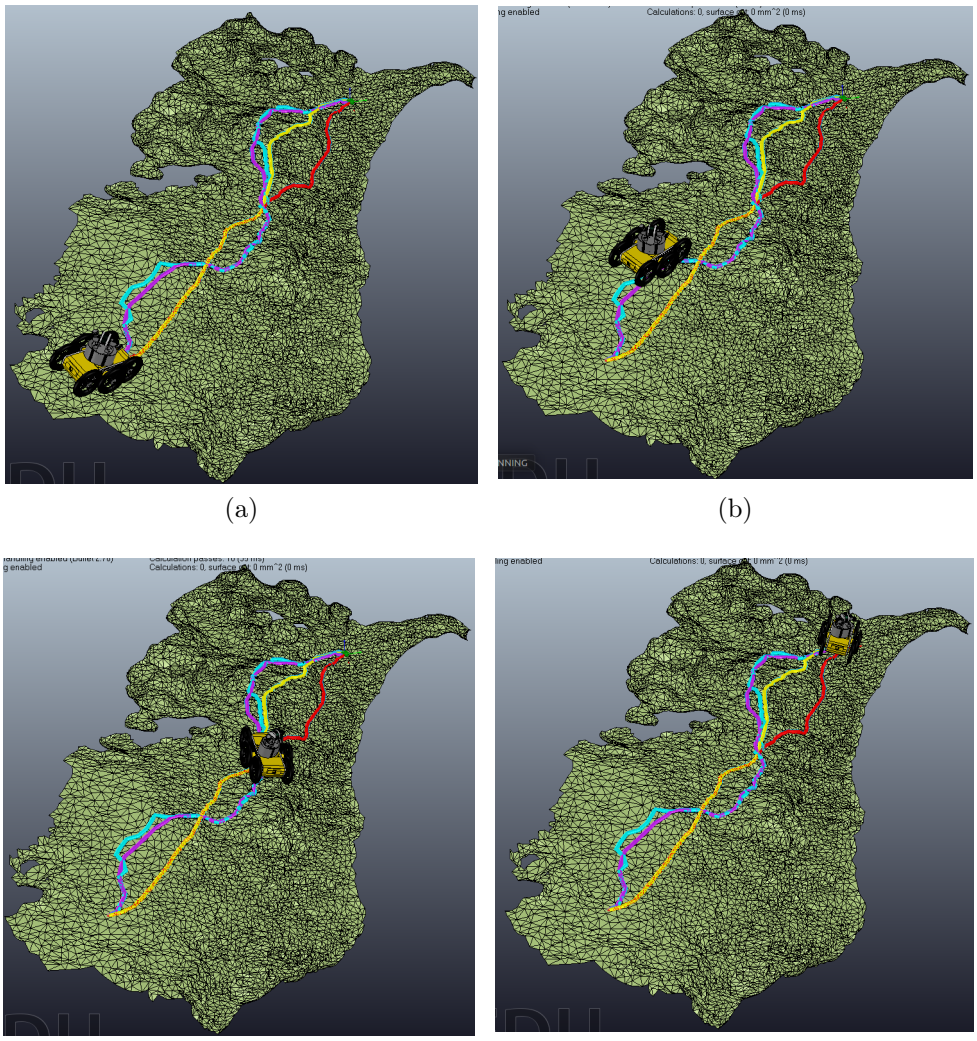
Figura 4.6: Caminhos ótimos obtidos utilizando o algoritmo de Dijkstra considerando as métricas de distância percorrida, transversalidade do terreno e consumo energético do robô em um mapa de cavidade natural considerando a cinemática do mesmo. As cores das malhas estão relacionadas com o eixo Z.

Tabela 4.5: Valores calculados para os caminhos ótimos obtidos nas simulações realizadas em um mapa de cavidade natural considerando a cinemática do robô.

Caminho	Distância percorrida (m)	Mínimo ângulo de tombamento (°)	Energia consumida (W)	Custo total normalizado
Mais curto	**8,33**	7,05	109800,00	23,25
Mais estável	15,26	**26,38**	282020,00	18,17
Mais econômico	8,48	18,70	**106120,00**	21,00
Métricas combinadas	11,43	23,50	182710,00	**16,19**

4.2.4 Simulações no V-REP utilizando um mapa de cavidade real

Uma vez obtidos os caminhos ótimos para cada métrica, foram realizadas simulações no V-REP para garantir que os caminhos são factíveis pelo robô. Para que o robô conseguisse completar todos os caminhos encontrados pelo algoritmo de Dijkstra, a velocidade linear foi mantida constante em 0,3m/s. A velocidade angular varia de forma proporcional à rotação que o robô precisa fazer para seguir o caminho desejado. Os coeficientes de atrito dinâmico e estático foram mantidos em 0,6 e 0,9, respectivamente. Como resultado, o robô conseguiu percorrer todos os caminhos encontrados pelo algoritmo de Dijkstra, não encontrando dificuldade. Uma das simulações está ilustrada pela Figura 4.7. Os caminhos são os mesmos apresentados na Figura 4.6. Um vídeo com algumas das simulações pode ser encontrado no *link*: `https://www.youtube.com/watch?v=YLUSqnO-z9I&feature=youtu.be`.



(a) (b)

(c) (d)

Figura 4.7: Seguimento de caminho em um mapa de cavidade natural utilizando o simulador V-REP: posição inicial (a), posição dois (b), posição três (c) e posição final (d).

4.2.5 Análise dos resultados considerando o modelo cinemático do robô

Novamente, os caminhos ótimos obtidos na simulação em um mapa plano sem ondulação foram os mesmos para todas as métricas. Isso devido à modificação realizada no algoritmo de Dijkstra descrita na Seção 3.1. Ainda, analisando os resultados das Tabelas 4.4 e 4.5, é possível concluir que os caminhos obtidos pelo algoritmo de Dijkstra considerando as métricas propostas mostraram ser ótimos, visto que os melhores valores de distância percorrida, ângulo mínimo de tombamento e energia consumida estão relacionados aos trajetos que utilizam as respectivas métricas no planejamento de caminho. O menor custo total normalizado também está relacionado com o caminhos com métricas combinadas. É mais intuitivo chegar a essa conclusão de forma visual observando os resultados obtidos na etapa de validação do algoritmo, ilustrados na Figura 4.4. Ainda, é possível observar que os caminhos encontrados utilizando as métricas de forma combinada alcançaram valores intermediários de distância percorrida, mínimo ângulo de tombamento e energia consumida.

Analisando os resultados apresentados na Tabela 4.5, o caminho com métricas combinadas é que o apresenta o menor custo total normalizado. Isso se deve à tentativa do algoritmo em minimizar todos os atributos de forma simultânea, levando em consideração o peso atribuído a cada uma.

4.3 Simulações realizadas com o Pybullet

Para realizar as simulações com o Pybullet, primeiramente é necessário importar o mapa do terreno e o modelo do robô, ambos em formato urdf. As variáveis de entrada utilizadas nas simulações são as coordenadas de posição (x, y e z) e o ângulo de guinada do robô.

Alguns parâmetros como os coeficientes de atrito, de amortecimento e de dureza do contato foram configurados de modo que o robô não deslize facilmente sobre o terreno.

Durante a simulação, o robô é solto de uma determinada altura, que inicialmente recebe o valor de z e é ajustada de modo que o robô esteja livre no ar, ou seja, sem nenhum ponto de contato com o terreno. A simulação termina quando:

- o chassi do robô colide com o terreno;
- o robô capota;
- o robô sai do mapa;
- o robô se afasta muito do ponto onde é solto.

Caso as condições de simulação são satisfeitas, o simulador retorna os seis pontos de contato do robô com o terreno e a posição do centro de massa do mesmo, e com esses dados são calculados os ângulos de tombamento do robô.

Um detalhe importante é que dependendo da configuração do robô em relação ao terreno, nem sempre todas as seis rodas estão em contato com o mesmo. Nesse caso, o simulador retorna os pontos mais próximos de cada roda do robô sobre o mapa.

Capítulo 5

Conclusões e trabalhos futuros

Nesta dissertação é apresentado um estudo sobre técnicas de planejamento de caminho para robôs móveis em ambientes acidentados, que servirá como base para o desenvolvimento de um sistema de navegação autônomo para o EspeleoRobô. Esse sistema terá como objetivo otimizar a exploração do robô em terrenos acidentados e garantir a integridade do equipamento.

Com os resultados obtidos, foi concluído que é possível realizar o planejamento de caminho utilizando as métricas propostas de distância percorrida, transversalidade do terreno e consumo energético do robô tanto considerando o mesmo como uma partícula quanto considerando seu modelo cinemático. A nova métrica de energia utiliza valores realistas obtidos por meio de simulações, enquanto a nova métrica de transversalidade mostrou ser eficaz visto que o robô conseguiu executar todas as trajetórias sem que houvesse capotamento. Ainda, é proposta uma função de custo que considera múltiplos objetivos durante o planejamento de caminho. Dessa maneira, o algoritmo avalia a relevância de cada métrica ao encontrar trajetos ligando as posições inicial e final.

Para trabalhos futuros, os códigos desenvolvidos em Matlab serão adaptados para o sistema que controla o EspeleoRobô, *Robot Operating System* (ROS). Ainda, serão estudadas técnicas de mapeamento e localização simultânea (SLAM) e algoritmos de exploração. Dessa forma, o sistema fará com que o robô seja capaz de mapear e se localizar no terreno, e escolher de forma automática os destinos que deseja explorar. Posteriormente, serão realizados testes em campo para avaliar o desempenho do robô funcionando em modo autônomo.

5.1 Publicações e apresentações

Durante a elaboração da dissertação foram desenvolvidos trabalhos no sentido de contribuir com a comunidade acadêmica em relação ao tema robótica:

- Sistema de Navegação para uma Cadeira de Rodas Elétrica (SANTOS *et al.*, 2017).

Apresentação de artigo no XIII Simpósio Brasileiro de Automação Inteligente (SBAI) 2017 em Porto Alegre.

- *Path Planning for Mobile Robots on Rough Terrain* (SANTOS *et al.*, 2018).
  Apresentação do artigo contendo os resultados preliminares desta dissertação, publicado e apresentado no 15th Latin American Robotics Symposium (LARS) 2018 em João Pessoa.

- *Multi-terrain inspection robotic device and methods for configuring and guiding the same* (FREITAS *et al.*, 2018).
  Participação no pedido de patente do EspeleoRobô depositado no *United States Patent and Trademark Office*.

5.2 Sugestões de trabalhos futuros

Nesta dissertação é feita uma prova de conceito para validação das metodologias propostas. Em trabalhos futuros, pretende-se realizar simulações com outros algoritmos de exploração, por exemplo o RRT*. Também é sugerido que os códigos desenvolvidos em Matlab sejam otimizados e adaptados para o sistema que controla o EspeleoRobô, *Robot Operating System* (ROS). Ainda, é sugerido o estudo de técnicas de mapeamento e localização simultânea (SLAM) e algoritmos de exploração. Dessa forma, o sistema fará com que o robô seja capaz de mapear e se localizar no terreno, e escolher de forma automática os destinos que deseja explorar. Posteriormente, seriam realizados testes em campo para avaliar o desempenho do robô funcionando em modo autônomo.

Referências Bibliográficas

ACEVEDO, J., ARRUE, B., MAZA, I., et al.. "Cooperative large area surveillance with a team of aerial mobile robots for long endurance missions", *Journal of Intelligent & Robotic Systems*, v. 70, n. 1-4, pp. 329–345, 2013.

BELLMAN, R. "On a routing problem", *Quarterly of applied mathematics*, v. 16, n. 1, pp. 87–90, 1958.

BIANCHI, R. "Robótica". 2013. Disponível em: <`https://slideplayer.com.br/slide/1357065/`>. Acesso em: 13 mar. 2019.

BROGGI, A., CARDARELLI, E., CATTANI, S., et al.. "Terrain mapping for off-road Autonomous Ground Vehicles using rational B-Spline surfaces and stereo vision". Em: *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pp. 648–653. IEEE, 2013.

BROWN, R. "Surface area from a z-matrix". 2012. Disponível em: <`https://www.mathworks.com/matlabcentral/answers/35501-surface-area-from-a-z-matrix`>. Acesso em: 18 feb. 2019.

BURGARD, W., HEBERT, M., BENNEWITZ, M. "World modeling". Em: *Springer handbook of robotics*, Springer, pp. 1135–1152, 2016.

CHERIF, M. "Motion planning for all-terrain vehicles: a physical modeling approach for coping with dynamic and contact interaction constraints", *IEEE Transactions on Robotics and Automation*, v. 15, n. 2, pp. 202–218, 1999.

CHOSET, M. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

COTA, E. *Implementação e avaliação de ténicas de odometria aplicadas a um dispositivo robótivo móvel.* Tese de Mestrado, Universidade Federal de Ouro Preto, 2019.

COTA, E., TORRE, M., ROCHA, F., et al.. "Disposito de monitoramento remoto de cavidades - EspeleoRobô". Em: *XIII Simpósio Brasileiro de Automação Inteligente*, 2017.

COUMANS, E., BAI, Y. "Pybullet, a python module for physics simulation for games, robotics and machine learning", *GitHub repository*, 2016.

DARPA. "DARPA Subterranean Challenge Aims to Revolutionize Underground Capabilities". 2017. Disponível em: <`https://www.darpa.mil/news-events/2017-12-21`>. Acesso em: 23 jul. 2018.

DIJKSTRA, E. "A note on two problems in connexion with graphs", *Numerische mathematik*, v. 1, n. 1, pp. 269–271, 1959.

DUDEK, G., JENKIN, M. *Computational principles of mobile robotics*. Cambridge university press, 2010.

FANKHAUSER, P., BLOESCH, M., GEHRING, C., et al.. "Robot-centric elevation mapping with uncertainty estimates". Em: *Mobile Service Robotics*, World Scientific, pp. 433–440, 2014.

FERGUSON, D., MORRIS, A., HAEHNEL, D., et al.. "An autonomous robotic system for mapping abandoned mines". Em: *Advances in Neural Information Processing Systems*, pp. 587–594, 2004.

FRANCIS, B. A., MAGGIORE, M. *Flocking and rendezvous in distributed robotics*. Springer, 2016.

FREITAS, G., ROCHA, F., TORRE, M., et al.. "Multi-Terrain Inspection Robotic Device and Methods for Configuring and Guiding the Same". fev. 9 2018. BR Patent 2018050025.

GE, S., CUI, J. "New potential functions for mobile robot path planning", *IEEE Transactions on robotics and automation*, v. 16, n. 5, pp. 615–620, 2000.

GIESBRECHT, J. *Global path planning for unmanned ground vehicles*. Relatório técnico, Defence Research and Development Canada, 2004.

GONZÁLEZ, D., PÉREZ, J., MILANÉS, V., et al.. "A review of motion planning techniques for automated vehicles", *IEEE Transactions on Intelligent Transportation Systems*, v. 17, n. 4, pp. 1135–1145, 2015.

HANSELMAN, D., LITTLEFIELD, B. *Mastering MATLAB 6: a comprehensive tutorial and reference*. Pearson, 2001.

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

HOLCZER, B. “Pathfinding Algorithms Maze Runner”. 2018. Disponível em: <https://www.globalsoftwaresupport.com/pathfinding-algorithms/>. Acesso em: 18 feb. 2019.

HORNUNG, A., WURM, K. M., BENNEWITZ, M., et al.. “OctoMap: An efficient probabilistic 3D mapping framework based on octrees”, *Autonomous Robots*, v. 34, n. 3, pp. 189–206, 2013.

ISHIGAMI, G., NAGATANI, K., YOSHIDA, K. “Path planning for planetary exploration rovers and its evaluation based on wheel slip dynamics”. Em: *Robotics and Automation, 2007 IEEE International Conference on*, pp. 2361–2366. IEEE, 2007.

JEDDISARAVI, K., ALITAPPEH, R., GUIMARÃES, F. “Multi-objective mobile robot path planning based on A* search”. Em: *Computer and Knowledge Engineering (ICCKE), 2016 6th International Conference on*, pp. 7–12. IEEE, 2016.

KÜMMERLE, R., RUHNKE, M., STEDER, B., et al.. “A navigation system for robots operating in crowded urban environments”. Em: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 3225–3232. IEEE, 2013.

LAVALLE, S. M. “Rapidly-exploring random trees: A new tool for path planning”, 1998.

LI, S. “Markov random field models in computer vision”. Em: *European conference on computer vision*, pp. 361–370. Springer, 1994.

LIPMAN, Y. “Bounded distortion mapping spaces for triangular meshes”, *ACM Transactions on Graphics (TOG)*, v. 31, n. 4, pp. 108, 2012.

MANDOW, A., MARTINEZ, J. L., MORALES, J., et al.. “Experimental kinematics for wheeled skid-steer mobile robots”. Em: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1222–1227. IEEE, 2007.

MARTÍNEZ, J. L., MANDOW, A., MORALES, J., et al.. “Approximating kinematics for tracked mobile robots”, *The International Journal of Robotics Research*, v. 24, n. 10, pp. 867–878, 2005.

OTTONI, G. “Planejamento de trajetórias para robôs móveis”, *Projeto de Graduação em Engenharia de Computação–FURG, Rio Grande*, 2000.

PAPADOPOULOS, E., REY, D. “A new measure of tipover stability margin for mobile manipulators”. Em: *Proceedings of IEEE International Conference on Robotics and Automation*, v. 4, pp. 3111–3116. IEEE, 1996.

RABOIN, E., ŠVEC, P., NAU, D., et al.. "Model-predictive target defense by team of unmanned surface vehicles operating in uncertain environments". Em: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 3517–3522. IEEE, 2013.

RAJA, R., DUTTA, A., VENKATESH, K. "New potential field method for rough terrain path planning using genetic algorithm for a 6-wheel rover", *Robotics and Autonomous Systems*, v. 72, pp. 295–306, 2015.

ROCHA, F. *Análise de mobilidade de um dispositivo robótico para inspeção remota de cavidades.* Tese de Mestrado, Universidade Federal de Ouro Preto, 2018. Disponível em: <`http://www.repositorio.ufop.br/handle/123456789/10066`>.

ROCHA, F., FREITAS, G., VIEIRA, P., et al.. "ANALISE DE MOBILIDADE DE UM DISPOSITIVO ROBOTICO PARA INSPECAO REMOTA DE CAVIDADES". Em: *XIII Simpósio Brasileiro de Automação Inteligente*, 2017.

ROHMER, E., SINGH, S. P., FREESE, M. "V-REP: A versatile and scalable robot simulation framework". Em: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1321–1326. IEEE, 2013.

SANTOS, A., FREITAS, G., EUZÉBIO, T., et al.. "SISTEMA DE NAVEGACAO PARA UMA CADEIRA DE RODAS ELÉTRICA". Em: *XIII Simpósio Brasileiro de Automação Inteligente*, 2017.

SANTOS, A., AZPÚRUA, H., PESSIN, G., et al.. "Path Planning for Mobile Robots on Rough Terrain". Em: *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*, pp. 265–270. IEEE, 2018.

SARANLI, U., BUEHLER, M., KODITSCHEK, D. E. "Rhex: A simple and highly mobile hexapod robot", *The International Journal of Robotics Research*, v. 20, n. 7, pp. 616–631, 2001.

SCHMIDT, D., BERNS, K. "Climbing robots for maintenance and inspections of vertical structuresA survey of design aspects and technologies", *Robotics and Autonomous Systems*, v. 61, n. 12, pp. 1288–1305, 2013.

SCHUSTER, M., BRUNNER, S., BUSSMANN, K., et al.. "Towards Autonomous Planetary Exploration", *Journal of Intelligent & Robotic Systems*, v. 93, n. 3-4, pp. 461–494, 2019.

SECCHI, H. A. "Una introducción a los robots móviles", *Instituto de Automática–INAUT. Universidade Nacional de San Juan–UNSJ–Argentina. Edição: Agosto de*, 2008.

SICILIANO, B., SCIAVICCO, L., VILLANI, L., et al.. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

SIMMONS, R., HENRIKSEN, L., CHRISMAN, L., et al.. "Obstacle avoidance and safeguarding for a lunar rover". Em: *AIAA Forum on Advanced Developments in Space Robotics*, 1996.

SINGH, S., SIMMONS, R., SMITH, T., et al.. "Recent progress in local and global traversability for planetary rovers". Em: *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, v. 2, pp. 1194–1200. IEEE, 2000.

STENTZ, A. "Optimal and efficient path planning for partially-known environments". Em: *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pp. 3310–3317. IEEE, 1994.

TRÉMAUX, P. "École Polytechnique of Paris (X: 1876)". Em: *French engineer of the telegraph in Public conference, December*, v. 2, pp. 0980–603, 2010.

VITTIN. "A-Star". 2017. Disponível em: <`https://github.com/vittin/A-Star`>. Acesso em: 18 feb. 2019.

ZIEGLER, A., GILBERT, D., MORSE, C., et al.. "Autonomous surface cleaning robot for wet and dry cleaning". mar. 5 2013. US Patent 8,387,193.

Apêndice A

Publicações

Apresentação do artigo "Path Planning for Mobile Robots on Rough Terrain"no 15th Latin American Robotics Symposium (LARS) 2018 em João Pessoa.

2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)

Path Planning for Mobile Robots on Rough Terrain

Alexandre S. Santos[1], Héctor I. Azpúrua[2], Gustavo Pessin[3], and Gustavo M. Freitas[4]

***Abstract*— Recently, mobile robots have been used in several applications, such as inspection, surveillance, exploration of unknown environments, among others. For example, the Vale's speleology group acquired a robotic platform, named EspeleoRobô, capable of moving on rough terrain, exploring and mapping natural caves. Although EspeloRobô achieved its goal in several field tests, we have faced repeated operational challenges regarding loss of communication between the control base and the robot. A possible solution is to implement autonomous navigation. Thus, an embedded instrumentation and control system would guide the robot to improve the exploration and prevent it from damage. Therefore, this paper evaluates path planning techniques to optimize the robot exploration on rough terrain. We represent the environment using triangle meshes modeled as a graph and generate optimal paths with the Dijkstra's algorithm considering the following metrics as cost function: traveled distance, terrain traversability, and robot power consumption. Lastly, this paper proposes a cost function that accounts for multiple goals during the path planning, and the user can set the trade-off between them through weights. Thus, the algorithm evaluates the relevance of each metric while finding paths connecting the start and goal positions.**

I. INTRODUCTION

One of the greatest challenges in robotics is to develop robots capable of performing tasks autonomously. These robots can be used in many applications such as inspection and maintenance [1], surveillance [2], urban transport [3], safety and defense [4], cleaning [5], among others.

Mobile robots are also employed to explore unknown environments, especially places that are harmful to human life, for instance, planetary exploration [6] and mapping of abandoned mines [7]. As an example, mining companies need to inspect natural caves to collect data for environmental studies and determine their relevance. The speleologists are the professionals in charge of making topographic surveys and identification of living beings in the interested areas. However, most of these caves are narrow and difficult to get to, where we can find several hazards, for example, landslides, and presence of wild and venomous animals. The mentioned conditions are shown in Fig. 1.

[1]Alexandre Souza Santos is with Escola de Minas, Universidade Federal de Ouro Preto (UFOP), Ouro Preto, MG, Brazil alexandre2525@gmail.com

[2]Héctor Ignacio Perez Azpúrua is with Instituto Tecnológico Vale (ITV), Ouro Preto, MG, Brazil hector.azpurua@itv.org

[3]Gustavo Pessin is with Instituto Tecnológico Vale (ITV), Ouro Preto, MG, Brazil gustavo.pessin@itv.org

[4]Gustavo Medeiros Freitas is with Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, MG, Brazil gustavo.medeiros.freitas@pq.itv.org

Fig. 1: Hazards related to the inspection of natural caves.

In order to support the speleologists' work, the Vale's speleology group started the project EspeleoRobô [8]-[9], acquiring a teleoperated robotic platform capable of moving on rough terrain, exploring and mapping natural caves. Although EspeloRobô achieved its goal in several field tests, we have faced repeated operational challenges regarding loss of communication. These failures could damage the robot if it falls into a hole or cause the loss of the equipment if it enters an area we cannot rescue. A solution is to implement autonomous navigation. Thus, an embedded instrumentation and control system would guide the EspeleoRobô to improve the exploration and prevent it from damage. The mapping is done through colored 3-D point clouds using high-resolution cameras and the sensor Velodyne LiDAR VLP-16. The tower's 3-D modeling coupled to the robot and a test field example are illustrated in Fig. 2.

(a) (b)

Fig. 2: (a) Instrumentation tower's 3-D modeling; (b) field test example. Retrieved from [10].

Regarding mobile robots, autonomous navigation involves mapping and localization, path planning, and motion control. This work aims to analyze path planning techniques to optimize the EspeleoRobô exploration on rough terrain. We represent the environment using triangle meshes modeled as a graph and generate optimal paths with the Dijkstra's algorithm considering the following metrics as cost function: traveled distance, terrain traversability, and robot power consumption. Finally, this paper proposes a cost function that accounts for multiple goals during the path planning, and the user can set the trade-off between them through weights. Therefore, the algorithm evaluates the relevance of each metric while finding navigable paths.

978-1-5386-7761-2/18/$31.00 ©2018 IEEE
DOI 10.1109/LARS/SBR/WRE.2018.00056

II. PATH PLANNING FOR MOBILE ROBOTS

Path planning for mobile robots has been studied for decades. This problem can be divided into two steps. The first one consists in representing the navigation environment, which is generally done using sensors such as cameras, lasers, sonar, among others. The second step is to utilize search algorithms to find an optimal path to the robot from its start position to the goal, being both defined by the user. Different routes can be reached depending on search criteria, like shortest traveled distance or lowest power consumption.

A. Models for Natural Environments

According to [11], the main geometric models used to represent natural environments are elevation grids, 3-D grids, and meshes.

Elevation grids models describe the terrain as a function $h = (x, y)$, where x e y represent the grid cells' planar coordinates, and h is the corresponding elevation. Due to their features of simple data structures, elevation maps can be easily generated from sensor data and have been widely used for mobile robots operating in natural environments [12]-[13]. Theses maps assume a 2-D plane as a reference direction. However, in many cases, this assumption cannot be assumed. A possible solution is to represent the data directly in 3-D grids [14], without projecting it onto a reference 2-D plane. Therefore, the data is kept in its original format, and there is no restriction on the geometry of the environment. Although 3-D grids are generic representations, they are costly computationally and do not clearly represent surface continuity. Another alternative is to model the terrain by meshes [15]. These models are also compact and can, in theory, represent any combination of surfaces. However, extracting the correct surfaces from raw data is not trivial in complex environments.

B. Graph Search Algorithms

Once the environment is represented by any geometric models described in Section 2.1, we need to find an optimal path for the robot from its start position to the goal, being both defined by the user. These models can be treated as graphs, which are structures composed of nodes or vertices connected by arcs or edges. We can attribute costs to the arcs and use a algorithm to find the lowest cost path between two nodes. The most common graph search algorithms found in the literature are Depth-First Search (DFS), Breadth-First Search (BFS), A*, Dijkstra and D*, and Rapidly-Exploring Random Trees (RRT).

The A* algorithm, initially proposed in [16], is one of the most employed search algorithms in robotics. This method combines two metrics into a cost function $f(n) = g(n) + h(n)$ to orient its search to find an optimal path. The first one, $g(n)$, is the attributed cost to move from the start position to the current one. The second, $h(n)$, is a heuristic that estimates the cost of the cheapest path from the current position to the goal. The algorithm always chooses the successor with the smallest cost function. A* is optimal if it has an admissible heuristic, which means that it never overestimates the cost $h(n)^*$ of the optimal path, that is, $h(n) \leq h(n)^*$. Under this premise, A* is capable of exploring routes with lower costs than the optimal path, only finishing the search when the algorithm finds the optimal path. Moreover, A* provides the fastest search of any other search algorithm which uses the same heuristic [17].

The Dijkstra's algorithm [18] is a special case of A* where $h(n) = 0$, that is, $f(n) = g(n)$. Thus, the method does not use a search heuristic to find an optimal path, using only the route cost until the algorithm finds the final position [19]. This algorithm is employed to find the lowest cost path between vertices of a graph with positive weights. Once a source node is chosen to start the search, the algorithm computes the lowest costs from this vertice to the remaining ones. The Dijkstra's algorithm has an initial estimative for the lowest cost route and increases it successively. A node is defined as closed when the algorithm had already found a path with the minimal cost from the source vertice to the current one. Otherwise, the node is in open state. When all vertices have reached closed state, the results are the lowest costs from the source node to the remaining ones. Thus, the algorithm finds the optimal paths using the vertices called predecessors.

The algorithms mentioned previously assume that the environment is fully known. Otherwise, the D* algorithm can be employed [20]. This method is used for partially known environments, static or dynamic. The D* is based on A*, the difference is that A* uses fixed costs for maps' arcs and computes only the optimal path from the start position to the goal. However, in D* the arcs' costs can change according to the environment, and the algorithm computes all the possible paths between the initial and final positions, making it expensive computationally.

C. Related Work

Many studies address path planning for mobile robots. For example, in [21] the authors developed a navigation system for a planetary rover using binocular vision. In this work, they divided the navigation into two stages, global and local, using the algorithms D* and Morphin [22], respectively. The global navigation aims to guide the vehicle to its destination, while the local navigation cares about obstacle avoidance. For each terrain patch the Morphin algorithm determines the roll and pitch angles, roughness, and measures of certainty. Both the global and local algorithms evaluate the terrain traversability and vote using weights for the best actions for the robot to execute. For validation, the authors performed field tests with the rover Bullwinkle, where the vehicle traveled for 100m in outdoor environments at an average speed of 15 cm/s. The authors compared several experiments using different weights for the global and local algorithms. As a result, the greater the value is given to D*, the closer the vehicle gets to the obstacles. This generally results in shorter paths, however, it increases the chance of collisions. On the other hand, if D* is weighted much less than Morphin (less than five times), the robot sometimes is not capable of reaching the final destination.

In [6], the authors propose a path planning algorithm to a planetary rover that considers its dynamic mobility. This because the planet surfaces are usually composed of loose soil, where the wheel slippage can make the vehicle to get stuck. As the wheel slippage depends on factors such as vehicle speed, ground characteristics, and wheel-to-ground interaction, incorporating these issues into path planning is not trivial. However, the authors use the Dijkstra's algorithm for path planning considering a criteria function composed of three metrics: terrain roughness, terrain inclination, and path length. These metrics are normalized and the trade-off between them is done trough weights. Thus, the algorithm evaluates the relevance of each one while finding navigable paths. The authors carried out simulations and evaluated two candidate paths, where they used the same weights for the metrics, but with different thresholds for terrain inclinations. Both tracks presented good results, but the path with lower inclination thresholds is safer, although it is longer when compared to the route with higher restrictions. This work is similar to ours, however, we represent the environment using triangle meshes instead of elevation maps, and our algorithm also accounts for robot energy consumption.

Another interesting study is presented in [23]. The authors present a motion planning algorithm for a six-wheel rover on rough terrain. The algorithm uses potential fields to represent the environment, where the proposed function consists of attractive, repulsive, tangential, and gradient forces. The gradient force is a function of the rover's roll, pitch, and yaw angles, which are derived from the robot kinematic model. The algorithm tries to find safe paths avoiding routes with high gradient values. The authors attributed weights to the components of the potential field function, which are optimized using genetic algorithms. Thus, the motion planning algorithm is capable of finding optimal paths. The proposed method also evaluates the vehicle's wheel velocity to ensure stability and prevent wheel slippage. The authors performed simulations and compared routes using the metrics of energy consumption, wheel slippage, attractive force, and path length. They conclude that the proposed method is capable of finding better routes when compared with the traditional potential field method.

In [24], the authors propose a multi-objective path planning for mobile robots based on A* search. They focused on the application of planetary exploration, where the environment contains holes, hills, and rocks. Thus, the objectives are to minimize the difficulty, danger, elevation, and length of the path from a star to a goal position. The path must be as short as possible to minimize the time and energy consumption, and safe to avoid any trouble for the robot. The authors also used weights to set the trade-off between the objectives and carried out simulations on two different scenarios, confirming the applicability of the proposed path planning algorithm.

III. METHODOLOGY

The methodology adopted in this work consists in studying the main path planning methods for mobile robots and environment representations. Then, we the define environment and robot models, and the graph search algorithm employed in this work. We evaluate routes reached optimizing the traveled distance, terrain traversability, and robot power consumption. Lastly, we propose a cost function that considers multiple goals during the path planning, and the user can set the trade-off between them through weights. Thus, the algorithm evaluates the relevance of each metric while finding paths connecting the start and goal positions. More directly, we developed the following activities:

- define the most suited path planning techniques for natural caves;
- implement and validate the path planning techniques using Matlab;
- analyze paths reached optimizing the traveled distance, terrain traversability, and robot power consumption;
- propose a method to find routes optimizing multiple metrics simultaneously;
- analyze the simulations results.

IV. TECHNIQUES AND METRICS EMPLOYED

A. Environment Model

In this work, we represent the environment using triangle meshes modeled as a graph. We use a map generated through point clouds collected by the Vale's speleology group while visiting areas with natural caves. This map is relevant because it has similar characteristics to the areas which the company often needs to inspect. In Fig. 5 we can see an example of this representation.

B. Robot Model

For simplification purposes, we did not consider the EspeloRobô's kinematic model. However, we know that the robot is a six-wheel skid-steering vehicle that can move back and forth, and make pivot turns, meaning that EspeleoRobô is theoretically capable of following any collision-free path on the ground through a sequence of linear and angular movements. As proposed in [25], we also consider the robot as a point mass for path planning. In our case, the robot can move from a mesh's triangle center to another in its neighborhood. We define the meshes as nodes, and the arcs connecting the nodes are associated with fix costs regarding the used metric. In Fig. 3 we can see the vertices the robot can reach from a reference node.

C. Graph Search Algorithm

We chose the Dijkstra's algorithm for our methodology because this algorithm has guaranteed convergence for all cases, while A* is only optimal if the heuristic $h(n)$ is admissible. Regarding the traveled distance metric, a valid heuristic is the Euclidean distance from the current position to the goal. However, it is not trivial to estimate the heuristics for the metrics of terrain traversability and robot power consumption because they are very dependent on the terrain characteristics. Besides, we can easily use the Dijkstra's algorithm to optimize multiple metrics simultaneously. The method is described in Algorithm 1.

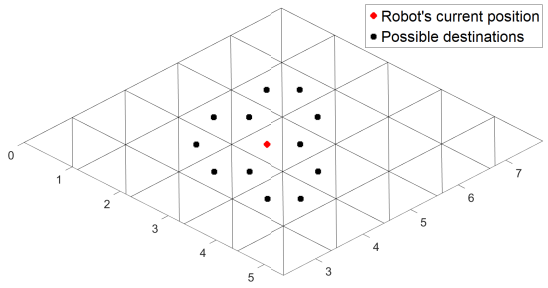


Fig. 3: Representation of the robot's current position and its possible destinations.

Algorithm 1: Pseudocode of Dijkstra's algorithm.

1: Add the initial node to the OPEN list and compute the cost function $f(n) = g(n)$.
2: Remove from the OPEN list the node with the smallest value of $f(n)$ and put it on the CLOSED list. This node has index n. In case of a tie, decide for the node with the smallest distance to the goal and with the smallest traveled distance.
3: **if** $n = goal$ **then**
4: Use the pointers to obtain the solution path.
5: END
6: **else**
7: Determine all the successor nodes (children) of n and compute the cost function for each successor not on the CLOSED list.
8: Associate with each successor node not on OPEN or CLOSED list the cost computed and put these on the OPEN list, placing pointers to n (n is the index of the parent node).
9: Associate with any successors already on OPEN list the smallest of the cost values just calculated and the previous cost value. (min(new $f(n)$, old $f(n)$))
10: Goto step 2.
11: **end if**

D. Traveled Distance Metric

This metric aims to find the shortest path from the start position to the goal, both defined by the user. The traveled distance is calculated by the following equation:

$$D_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}, \quad (1)$$

where D_{ij} is the Euclidean distance between the neighboring nodes of indexes i and j. The cost function using this metric is defined as follows:

$$C_1(p) = \sum_{n=p} D(n), \quad (2)$$

where the route p consists of a set of neighboring nodes given by $p = \{n_{initial}, ..., n_i, ..., n_{final}\}$. The shortest path p_{dist} is found by satisfying the following equation:

$$minC_1(p) = C_1(p_{dist}). \quad (3)$$

E. Terrain Traversability Metric

The terrain traversability metric intends to find the most flattest path from the start position to the goal. This metric is calculated as follows:

$$T_i = arccos\left(\frac{\left|\vec{N_i} \cdot \vec{Z}\right|}{\left\|\vec{N_i}\right\|\left\|\vec{Z}\right\|}\right), \quad (4)$$

where T_i is the positive sharp angle between the mesh normal vector ($\vec{N_i}$) and the Z-axis ($\vec{Z}$). The cost function using this metric is described by the following equation:

$$C_2(p) = \sum_{n=p} T(n), \quad (5)$$

where the route p consists of a set of neighboring nodes given by $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. The flattest path p_{tran}, whose angle sum is the smallest, is found as follows:

$$minC_2(p) = C_2(p_{tran}). \quad (6)$$

F. Energy Consumption Metric

This metric aims to find the most economical path from the start to the goal, both defined by the user. For simplification purposes, we considered that the robot moves with constant velocity. This means that the robot spends energy while breaking to keep movement uniformity. Thus, the energy metric is computed by the following equation:

$$E_{ij} = |\mu.m.g.cos\theta + m.g.sin\theta|.D_{ij}, \quad (7)$$

where E_{ij} is the energy that the robot spends to move from neighboring nodes of indexes i and j, μ is the friction coefficient, m is the robot mass, g is the gravity acceleration, and θ is the angle between the vector linking the mesh centers of indexes i and j, and the XY-plane. For simplification purposes, we chose the values $\mu = 1$, $m = 20kg$, and $g = 9,8m/s^2$. The cost function using this metric is described as follows:

$$C_3(p) = \sum_{n=p} E(n), \quad (8)$$

where the route p consists of a set of neighboring nodes given by $p = \{n_{inicial}, ..., n_i, ..., n_{final}\}$. The most economical path p_{ener} is found by the following equation:

$$minC_3(p) = C_3(p_{ener}). \quad (9)$$

G. Combined Metrics

We proposed a cost function that considers the multiple metrics mentioned previously during the path planning, and the user can set the trade-off between them through weights. Therefore, the algorithm evaluates the relevance of each metric while finding paths connecting the start and goal positions. Thus, the cost function is given as follows:

$$C_4(p) = \sum_{n=p}(P_d N_d D(n) + P_t N_t T(n) + P_e N_e E(n)), \quad (10)$$

where P_d,P_t e P_e are the weights that set the priorities of the metrics of distance traveled, terrain traversability, and robot power consumption, respectively. For example, if the priority is the exploration time, the user should set P_d with a value greater than the others. In environments with high risk of

TABLE I: Computed values for the optimal paths found during the simulations performed using a map with Gaussian curvatures.

Path	Traveled distance (m)	Angle sum (°)	Energy consumption (J)	Max Angle (°)
Shortest	7.43	1893.20	1671.20	56.58
Flattest	16.47	8.45	3233.20	7.42
Most economical	7.61	1143.90	1585.70	39.65
Combined metrics	8.45	506.68	1725.10	25.77

TABLE II: Computed values for the optimal paths found during the simulations performed using a natural cave map.

Path	Traveled distance (m)	Angle sum (°)	Energy consumption (J)	Max Angle (°)
Shortest	4.94	2692.80	1080.10	87.95
Flattest	6.14	1084.50	1362.00	40.04
Most economical	4.99	3928.30	1050.80	88.85
Combined metrics	5.48	1269.00	1222.40	47.96

tipping over, P_t requires a higher priority than the remaining weights. On the other hand, in situations where the energy consumption is critical, P_e should be greater than the others. The sum of these weights has unit value, resulting in a total of 100%. N_d, N_t e N_e are the normalization coefficients, which are calculated according to the neighboring nodes. The route p_{comb} is found by the following equation:

$$minC_4(p) = C_4(p_{comb}). \tag{11}$$

V. RESULTS

Initially, we created a 3D map represented by triangle meshes and composed of Gaussian curvatures (Fig. 4). We generated this map to visually validate the algorithm, which could not be intuitive on a rough terrain map. We set the weights $P_d = 0,25$, $P_t = 0,5$, and $P_e = 0,25$ to find an optimal path with combined metrics. Note that we set P_t with doubled value in regards to the other weights. Otherwise, the optimal path would be biased. This because both the metrics of traveled distance and energy consumption use the distance while computing their respective cost functions. In Fig. 4 we can see the optimal paths found for each metric. The values of traveled distance, angle sum, energy consumption, and max angle of each path are described in Table I.

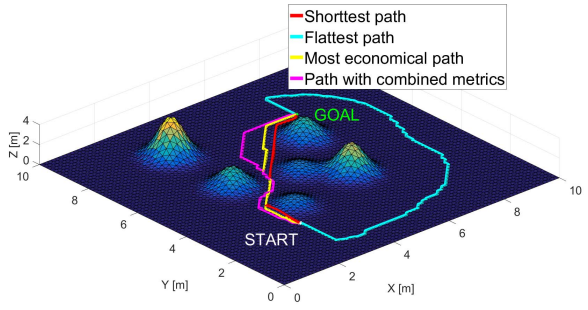


Fig. 4: Optimal paths found using the Dijkstra's algorithm optimizing the traveled distance, terrain traversability, and robot energy consumption on a map composed of Gaussian curvatures.

After completing the validation stage, we performed simulations using a natural cave map. We kept the same weights for these simulations (i.e., $P_d = 0,25$, $P_t = 0,5$, and $P_e = 0,25$). In Fig. 5 we can see the optimal paths found for each metric. The values of traveled distance, angle sum, energy consumption, and max angle of each path are presented in Table II.

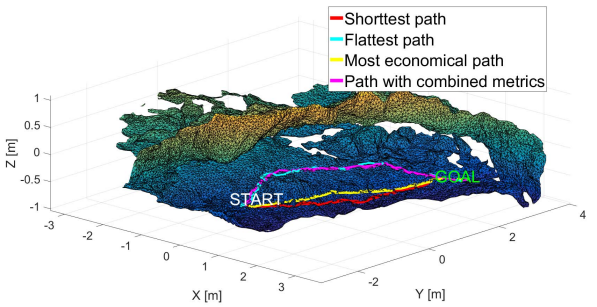


Fig. 5: Optimal paths found using the Dijkstra's algorithm to optimize the traveled distance, terrain traversability, and robot energy consumption on a natural cave map.

As expected, analyzing the results presented in Tables I and II, we conclude that the paths found using the Dijkstra's algorithm optimizing the proposed metrics are optimal, since the lowest values of traveled distance, angle sum, and energy consumption are related to the routes which use the corresponding parameters for path planning. It is easier to get to this conclusion by observing the results obtained in the validation stage, illustrated in Fig. 4. Furthermore, we can notice that the paths found using the combined metrics reached intermediate values of traveled distance, angle sum, and power consumption. Analyzing the results presented in Table II, the optimal path with combined metrics is only 10.93% longer than the shortest route, has the angle sum 17.01% greater than the flattest path, and is 16.33% less economic than the most economical path. This is because the algorithm tries to optimize all the parameters simultaneously, considering the weight of each one.

VI. CONCLUSIONS

In this work, we present a study of path planning techniques, which is the basis for the development of an autonomous navigation system for EspeleoRobo. This system aims to optimize the robot exploration on rough terrain and prevent it from damage.

With the reached results, we conclude that is it possible to perform path planning using the metrics of traveled distance, terrain traversability, and energy consumption. Moreover, we propose a cost function that considers multiple goals during the path planning. Thus, the algorithm evaluates the relevance of each metric while finding paths connecting the start and goal positions. Future works will compare the results obtained with the proposed metrics and other strategies described in literature for path planning based on mobility optimization.

Also, we will consider the EspeloRobô kinematic model and update the metrics of terrain traversability and energy consumption considering the tip-over angles and the costs of changing the robot direction, respectively. Thus, the path planning algorithm will guarantee that the paths are feasible and prevent the robot from tipping over. After finishing the simulations, we will adapt the codes developed in Matlab to run in Robot Operating System (ROS) embedded in the EspeleoRobô, allowing to perform field tests and evaluate the robot working in autonomous mode.

ACKNOWLEDGMENT

The authors would like to thank Instituto Tecnológico Vale - Brasil (ITV) and Universidade Federal de Ouro Preto - Brasil (UFOP) for providing the equipments and facilities. This study was partially financed by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and the Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (CNPq).

REFERENCES

[1] D. Schmidt and K. Berns, "Climbing robots for maintenance and inspections of vertical structures—a survey of design aspects and technologies," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1288–1305, 2013.

[2] J. Acevedo, B. Arrue, I. Maza, and A. Ollero, "Cooperative large area surveillance with a team of aerial mobile robots for long endurance missions," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1-4, pp. 329–345, 2013.

[3] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard, "A navigation system for robots operating in crowded urban environments," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3225–3232.

[4] E. Raboin, P. Švec, D. Nau, and K. Gupta, "Model-predictive target defense by team of unmanned surface vehicles operating in uncertain environments," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3517–3522.

[5] A. Ziegler, D. Gilbert, C. J. Morse, S. Pratt, P. Sandin, N. Dussault, and A. Jones, "Autonomous surface cleaning robot for wet and dry cleaning," Mar. 5 2013, US Patent 8,387,193.

[6] G. Ishigami, K. Nagatani, and K. Yoshida, "Path planning for planetary exploration rovers and its evaluation based on wheel slip dynamics," in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 2361–2366.

[7] D. Ferguson, A. Morris, D. Haehnel, C. Baker, Z. Omohundro, C. Reverte, S. Thayer, C. Whittaker, W. Whittaker, W. Burgard *et al.*, "An autonomous robotic system for mapping abandoned mines," in *Advances in Neural Information Processing Systems*, 2004, pp. 587–594.

[8] E. Cota, M. Torre, F. Rocha, G. Garcia, A. Junior, V. Ramos, V. Queiroz, V. Zanini, G. Brito, A. Marques, E. Pinto, L. Nogueira, G. Freitas, W. Miola, M. Reis, R. Araújo, and I. Brandi, "Disposito de monitoramento remoto de cavidades - EspeleoRobô," in *XIII Simpósio Brasileiro de Automação Inteligente*, 2017.

[9] G. Freitas, F. Rocha, M. Torre, A. Junior, V. Ramos, L. Nogueira, A. Santos, E. Cota, W. Miola, M. Reis, B. Costa, L. Ledezma, R. Evangelista, P. Alcantra, R. Lima, T. Souza, I. Brandi, R. Araujo, and M. Gomes, "Multi-terrain inspection robotic device and methods for configuring and guiding the same," Feb. 9 2018, BR Patent 2018050025.

[10] F. Rocha, G. Freitas, P. Vieira, M. Wilson, N. Ramon, and V. Iuri, "Analise de mobilidade de um dispositivo robotico para inspecao remota de cavidades," in *XIII Simpósio Brasileiro de Automação Inteligente*, 2017.

[11] W. Burgard, M. Hebert, and M. Bennewitz, "World modeling," in *Springer handbook of robotics*. Springer, 2016, pp. 1135–1152.

[12] A. Broggi, E. Cardarelli, S. Cattani, and M. Sabbatelli, "Terrain mapping for off-road autonomous ground vehicles using rational b-spline surfaces and stereo vision," in *Intelligent Vehicles Symposium (IV), 2013 IEEE*. IEEE, 2013, pp. 648–653.

[13] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *Mobile Service Robotics*. World Scientific, 2014, pp. 433–440.

[14] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[15] Y. Lipman, "Bounded distortion mapping spaces for triangular meshes," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 108, 2012.

[16] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[17] J. Giesbrecht, "Global path planning for unmanned ground vehicles," Defence Research and Development Canada, Tech. Rep., 2004.

[18] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[19] M. Choset, *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[20] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*. IEEE, 1994, pp. 3310–3317.

[21] S. Singh, R. Simmons, T. Smith, A. Stentz, V. Verma, A. Yahja, and K. Schwehr, "Recent progress in local and global traversability for planetary rovers," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 1194–1200.

[22] R. Simmons, L. Henriksen, L. Chrisman, and G. Whelan, "Obstacle avoidance and safeguarding for a lunar rover," in *AIAA Forum on Advanced Developments in Space Robotics*, 1996.

[23] R. Raja, A. Dutta, and K. Venkatesh, "New potential field method for rough terrain path planning using genetic algorithm for a 6-wheel rover," *Robotics and Autonomous Systems*, vol. 72, pp. 295–306, 2015.

[24] K. Jeddisaravi, R. Alitappeh, and F. Guimarães, "Multi-objective mobile robot path planning based on A* search," in *Computer and Knowledge Engineering (ICCKE), 2016 6th International Conference on*. IEEE, 2016, pp. 7–12.

[25] S. Ge and J. Cui, "New potential functions for mobile robot path planning," *IEEE Transactions on robotics and automation*, vol. 16, no. 5, pp. 615–620, 2000.

Participação no pedido de patente do EspeleoRobô depositado no *United States Patent and Trademark Office*.

FIGURES

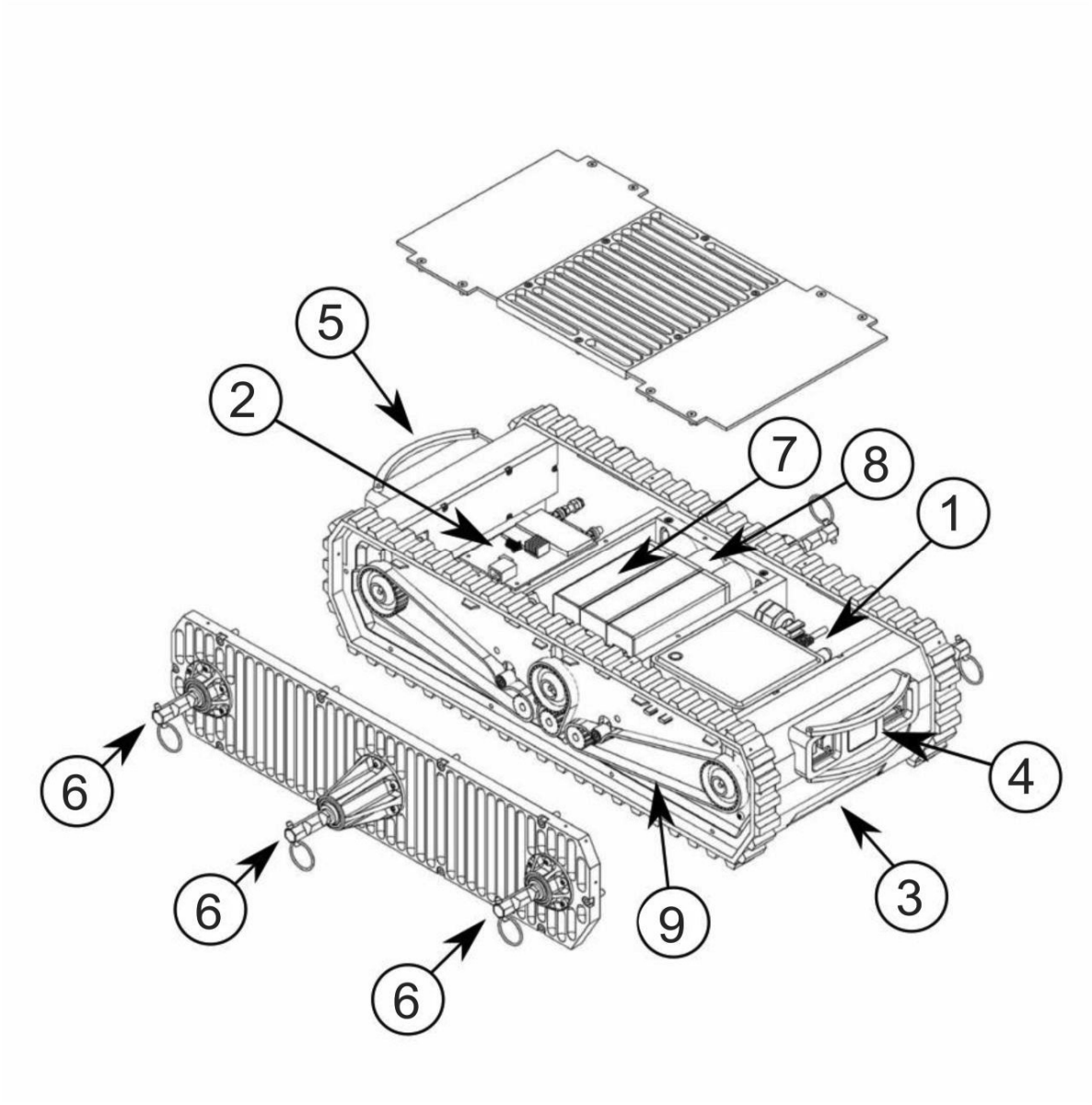


FIG. 1

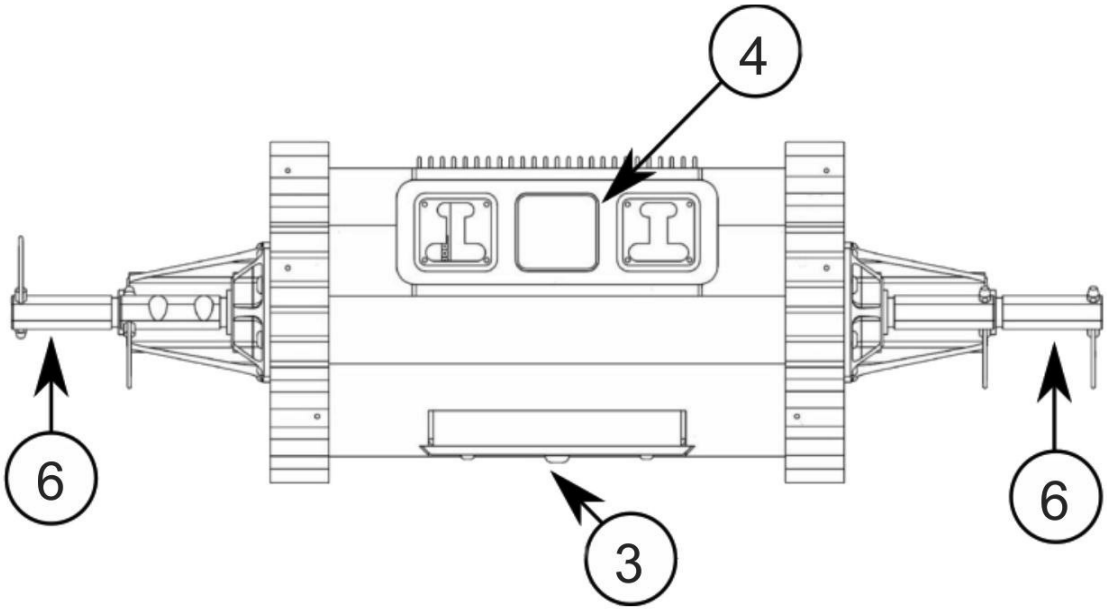


FIG. 2

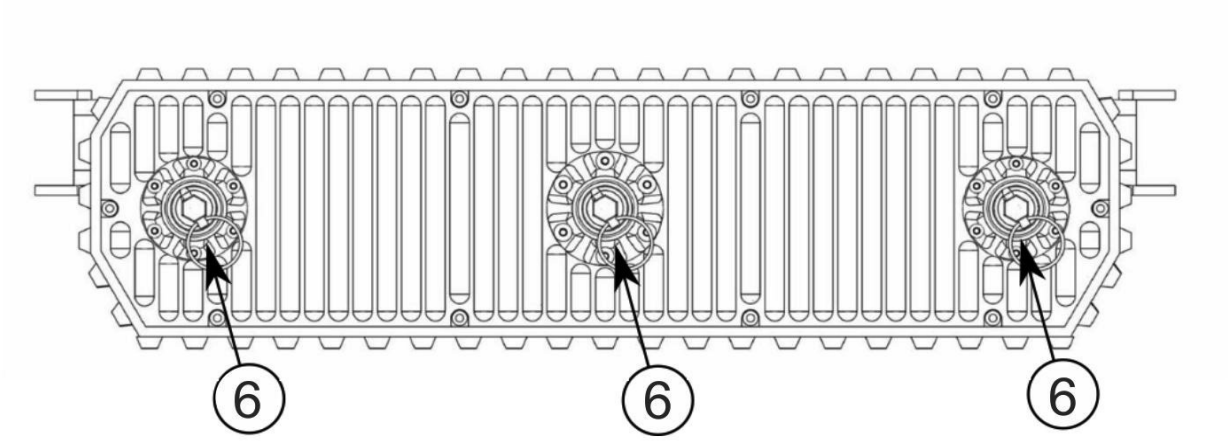


FIG. 3

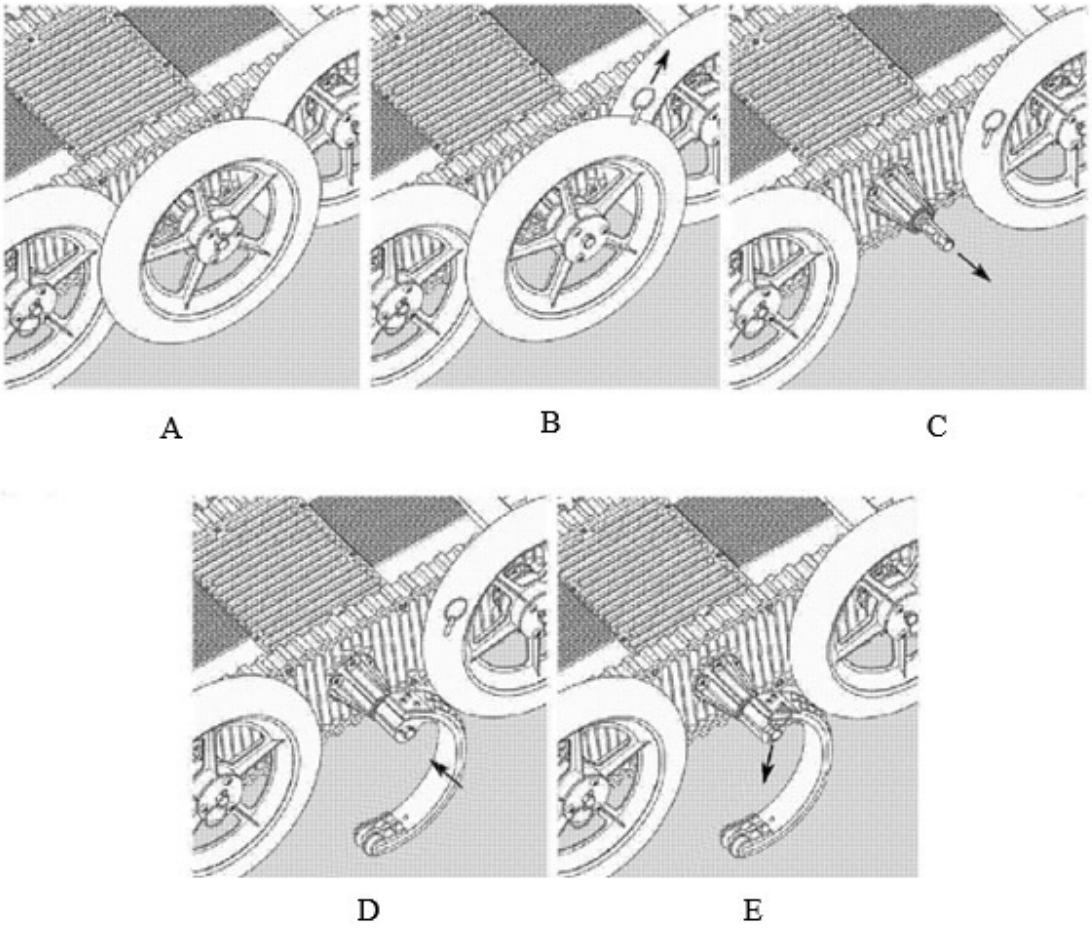


FIG. 4

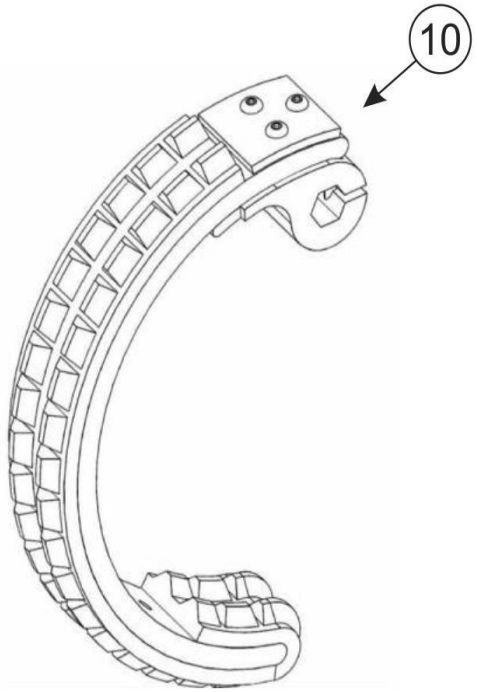


FIG. 5

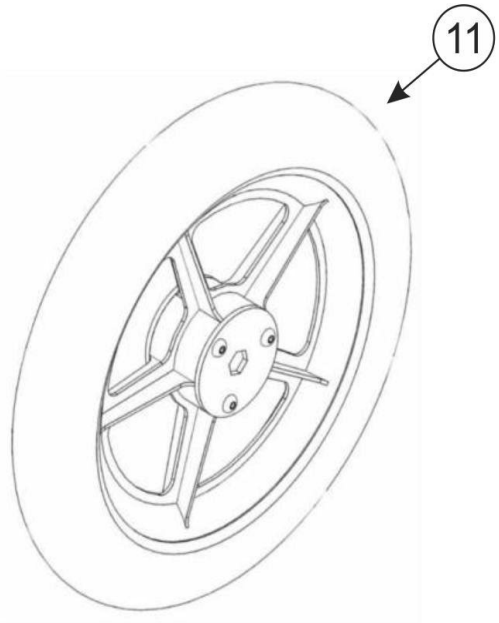


FIG. 6

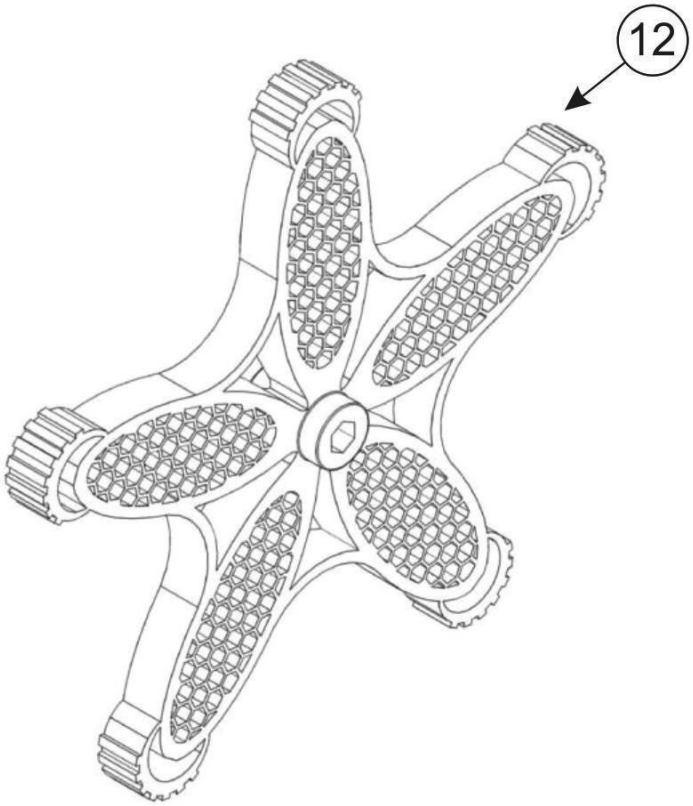


FIG. 7

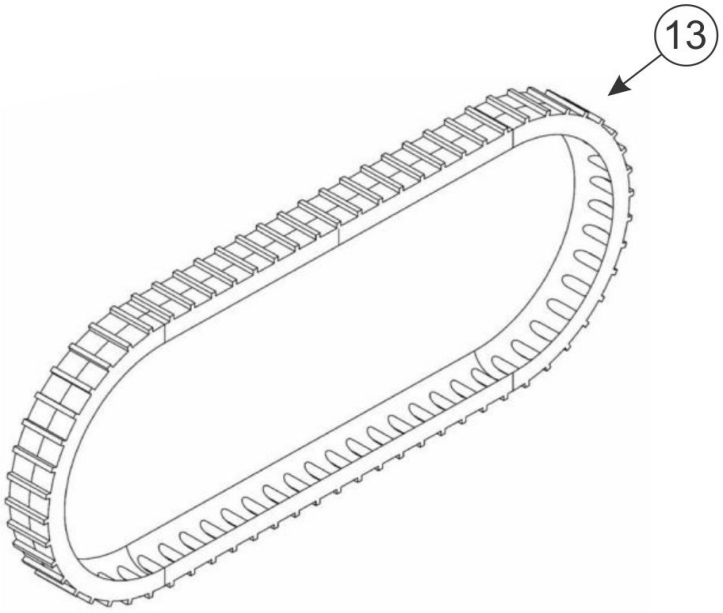


FIG. 8

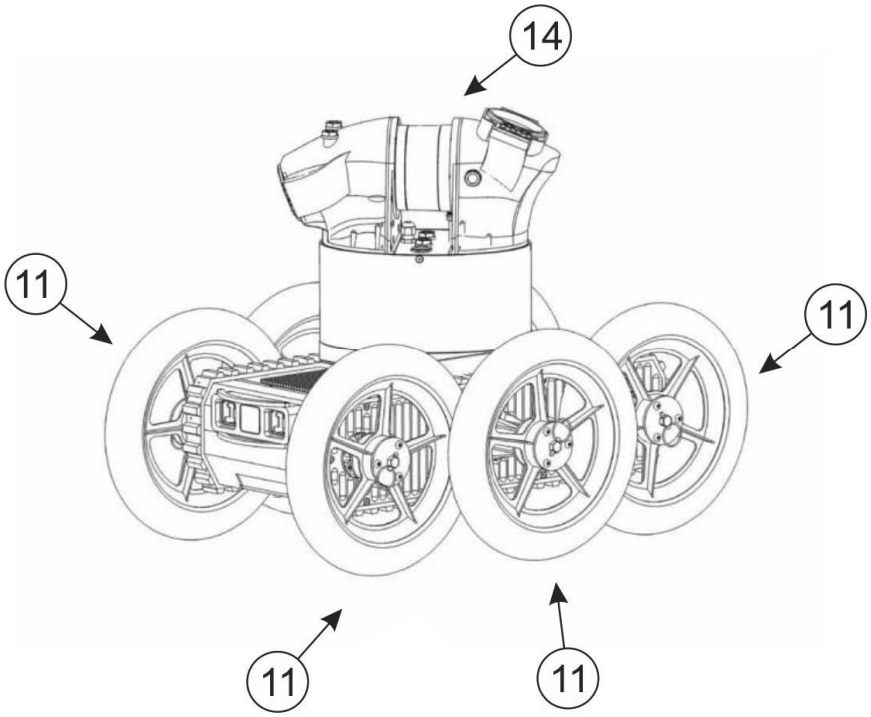


FIG. 9

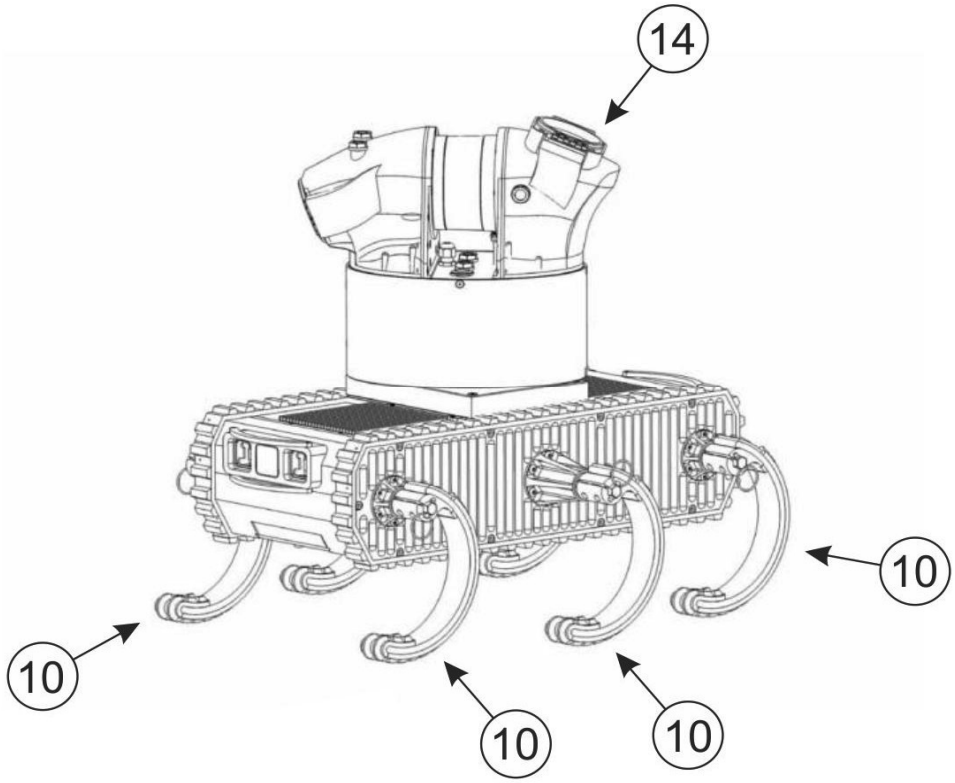


FIG. 10

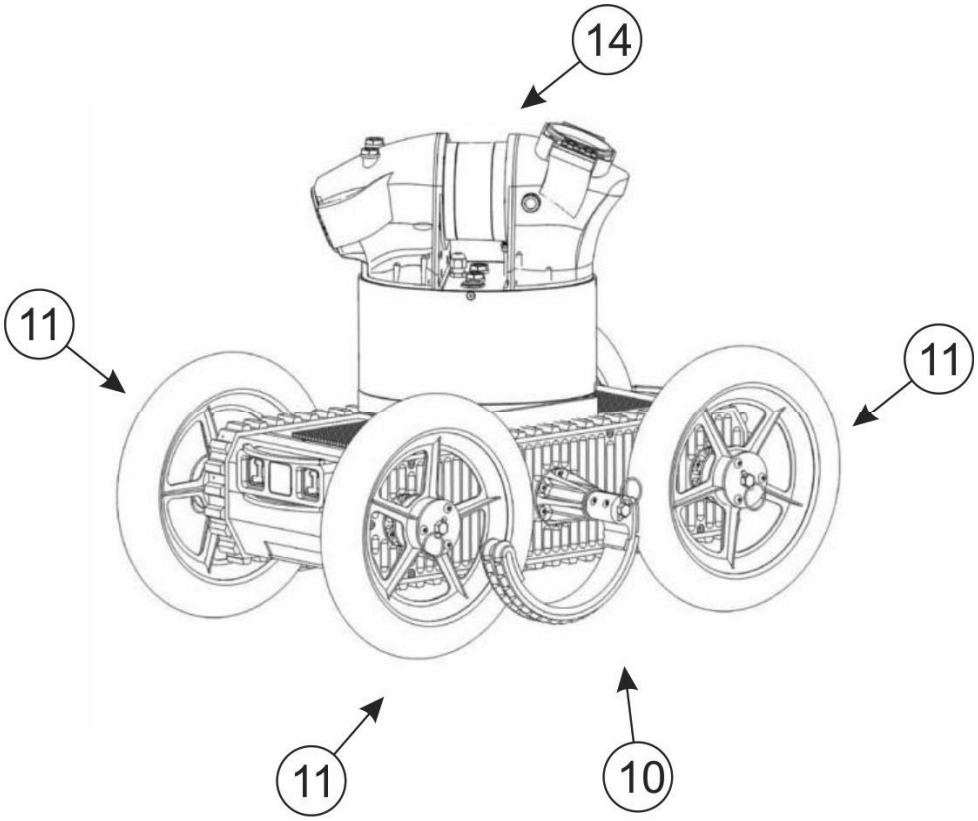


FIG. 11

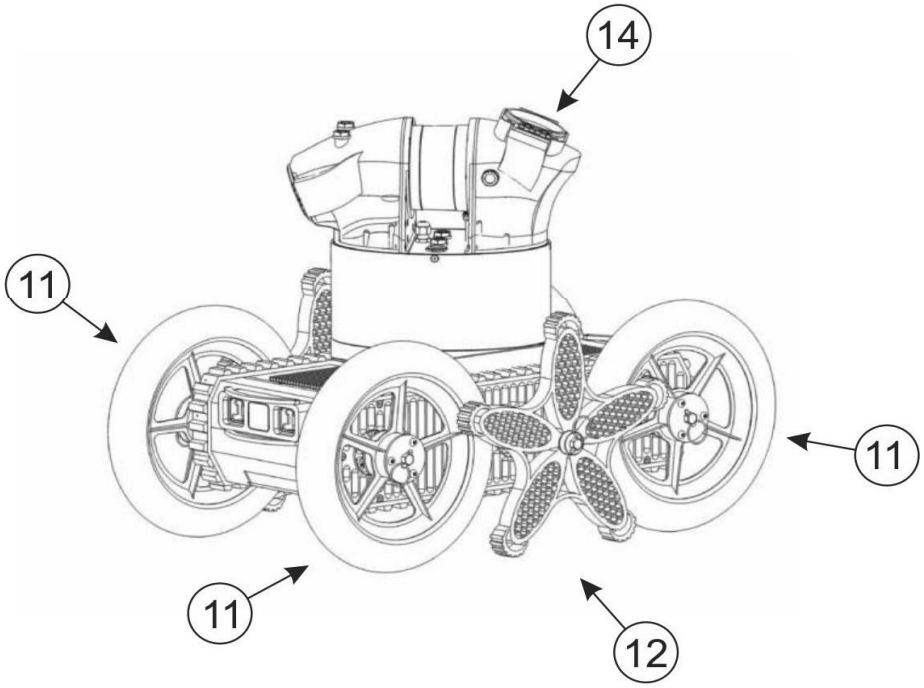


FIG. 12

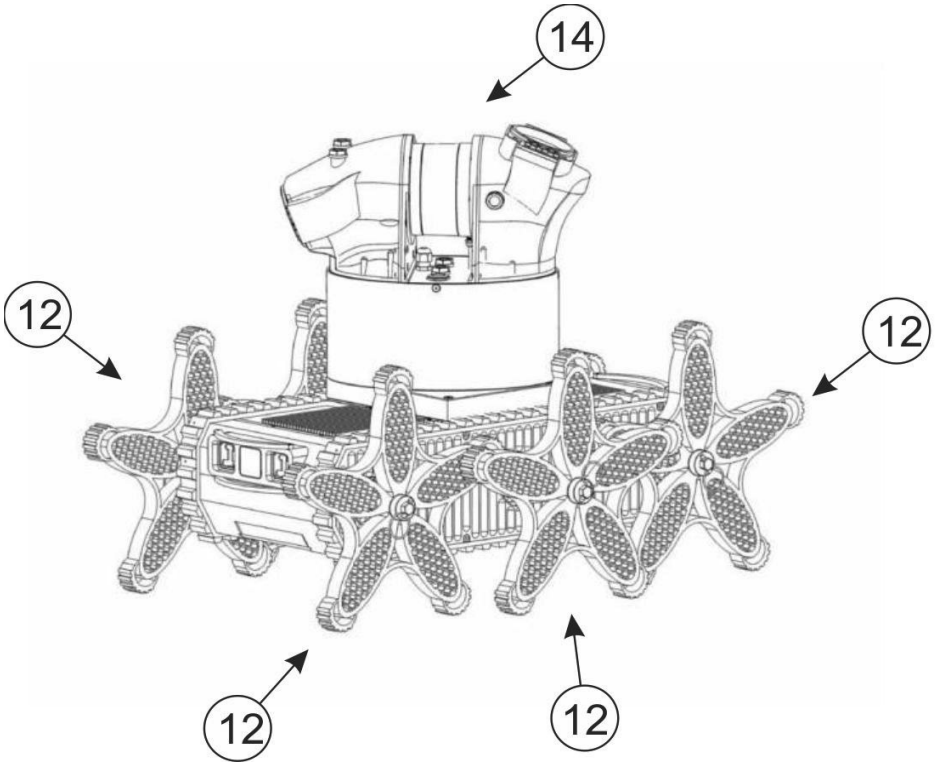


FIG. 13

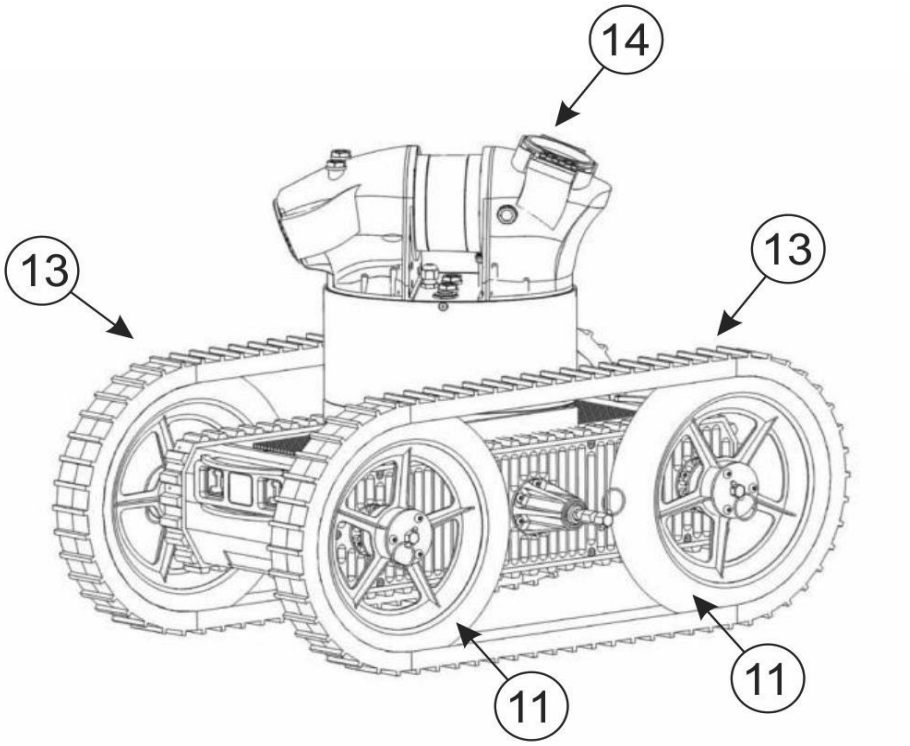


FIG. 14

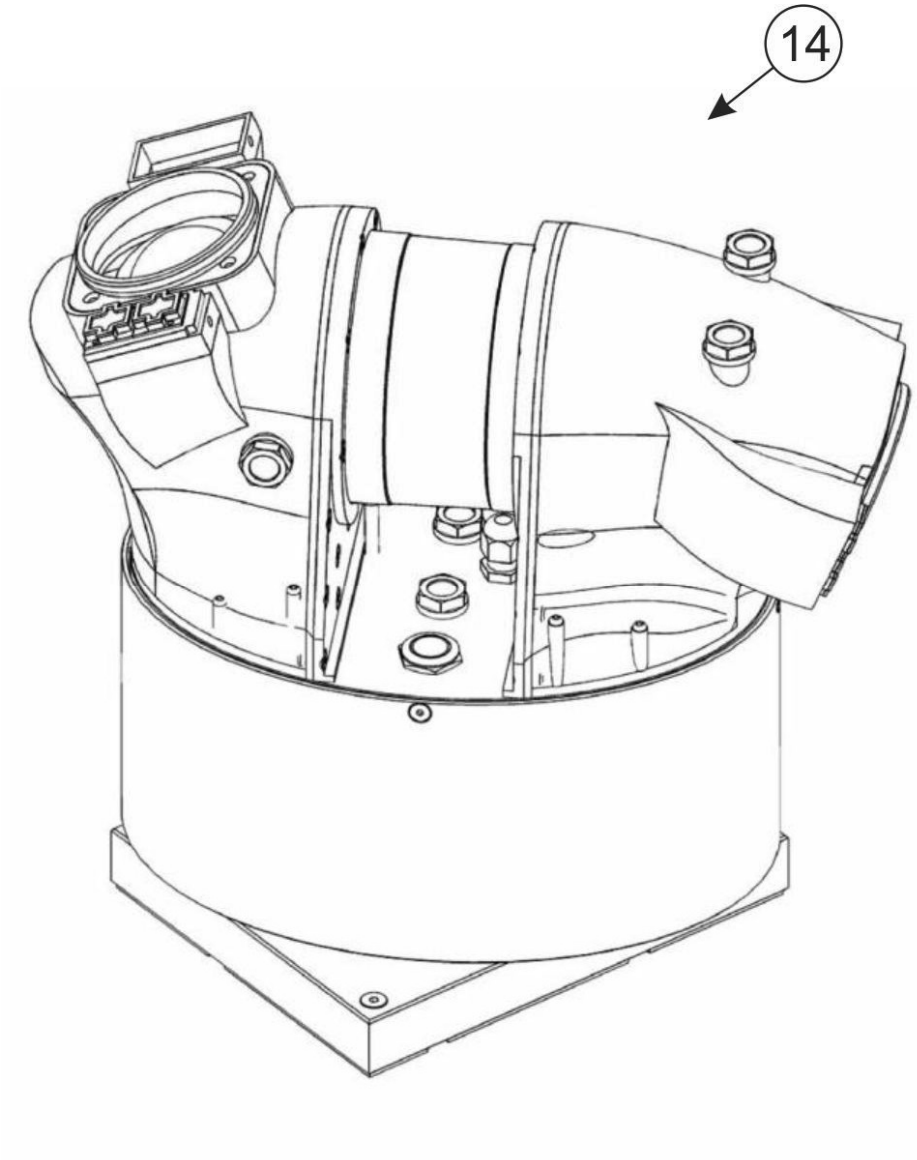


FIG. 15

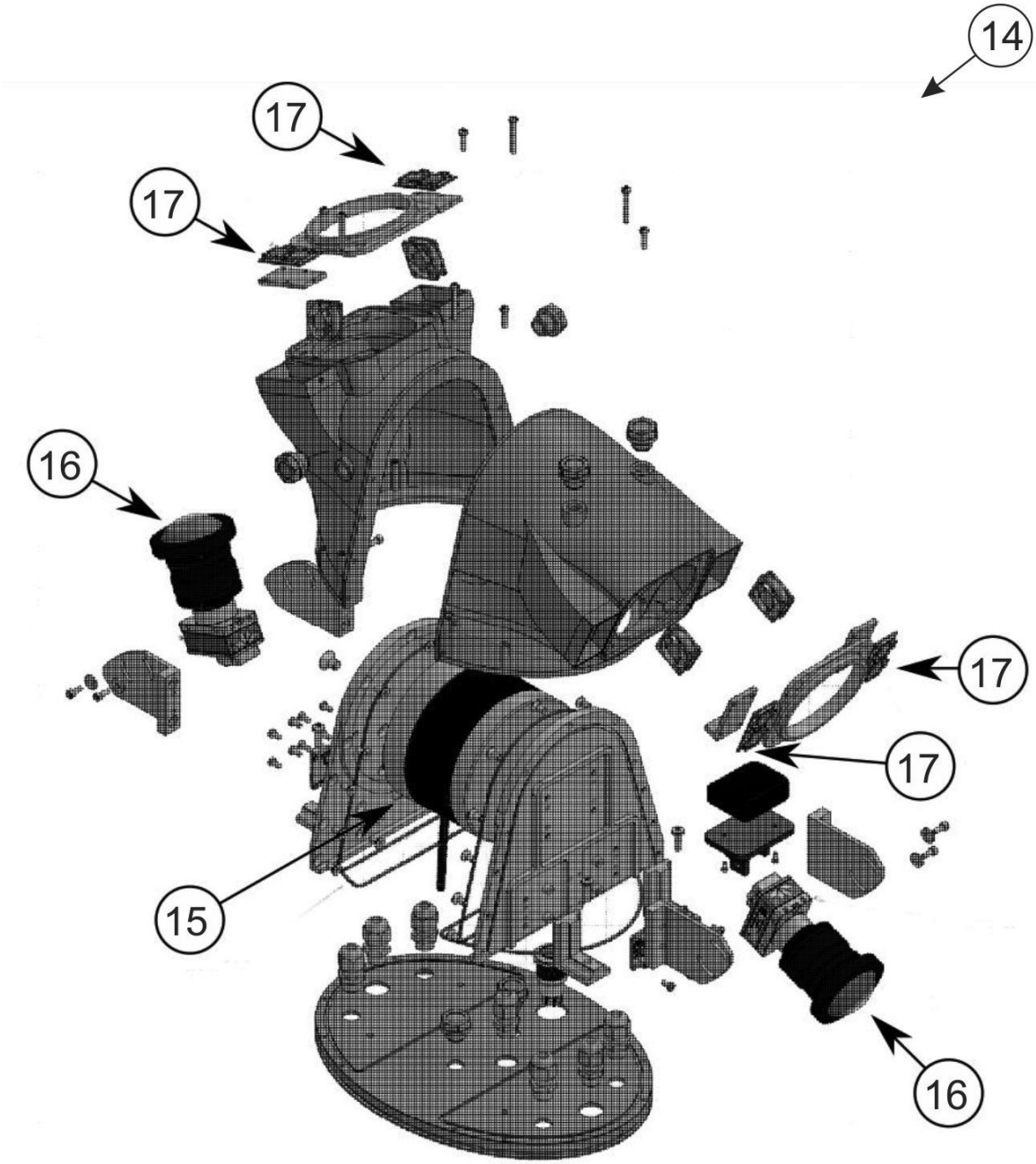


FIG. 16

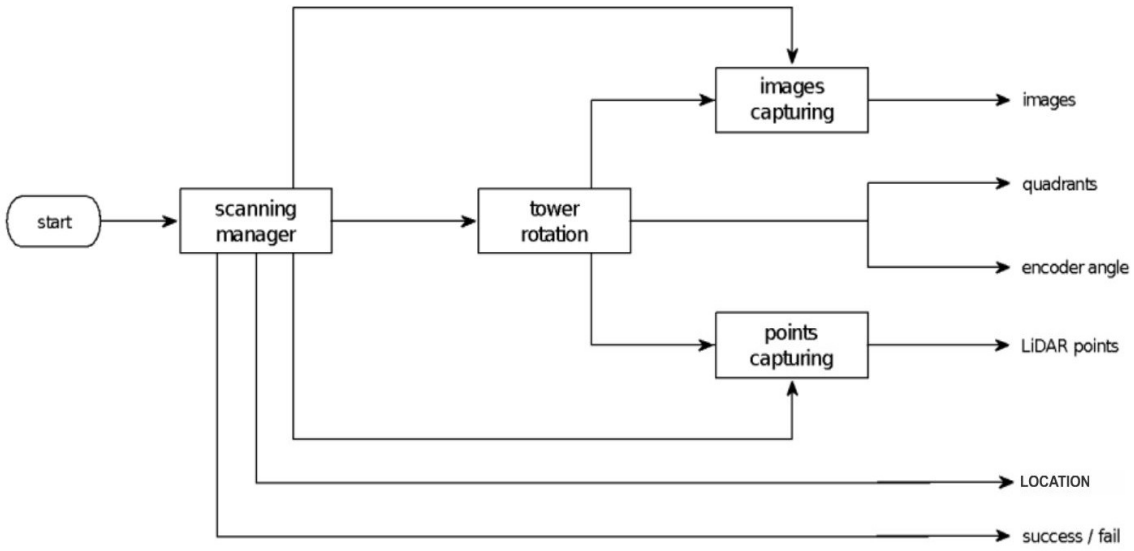

start
scanning manager
tower rotation
images capturing
points capturing
images
quadrants
encoder angle
LiDAR points
LOCATION
success / fail

FIG. 17

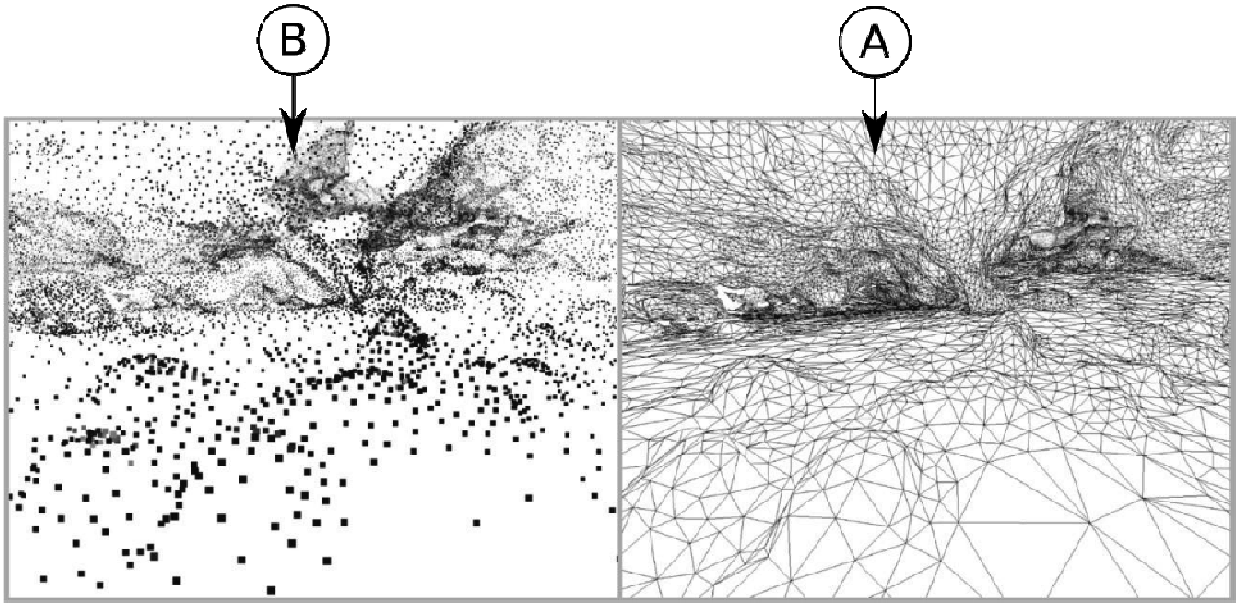

B
A

FIG. 18

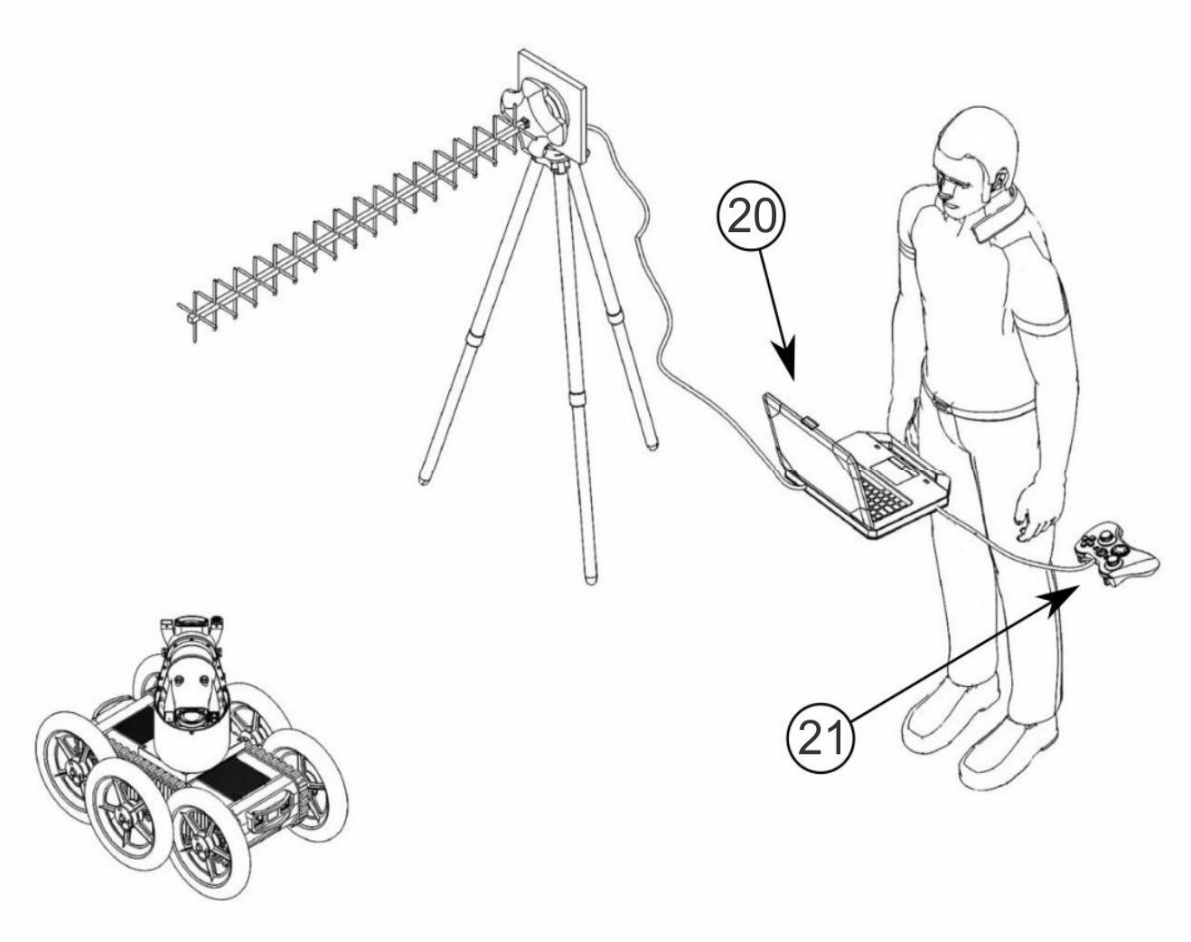


FIG. 19

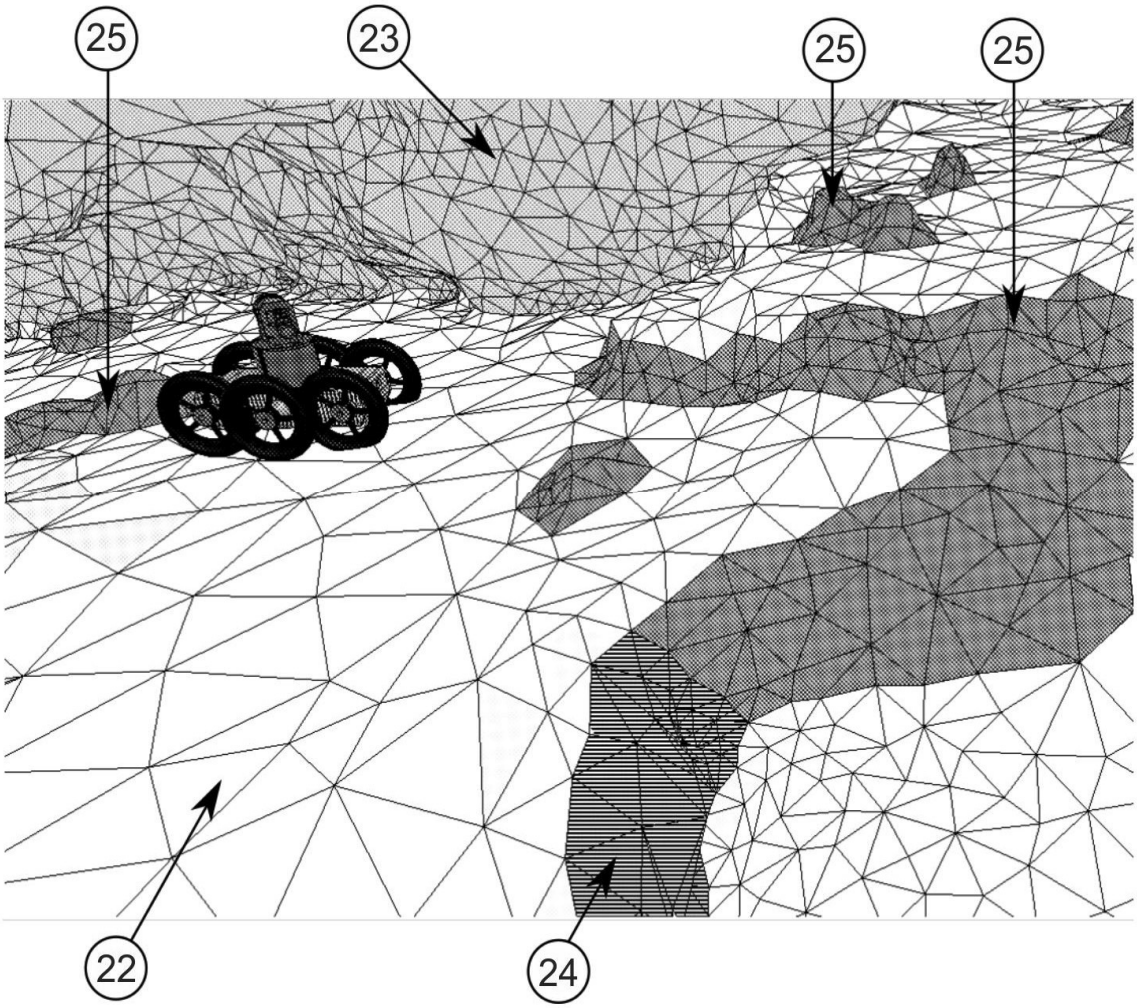


FIG. 20

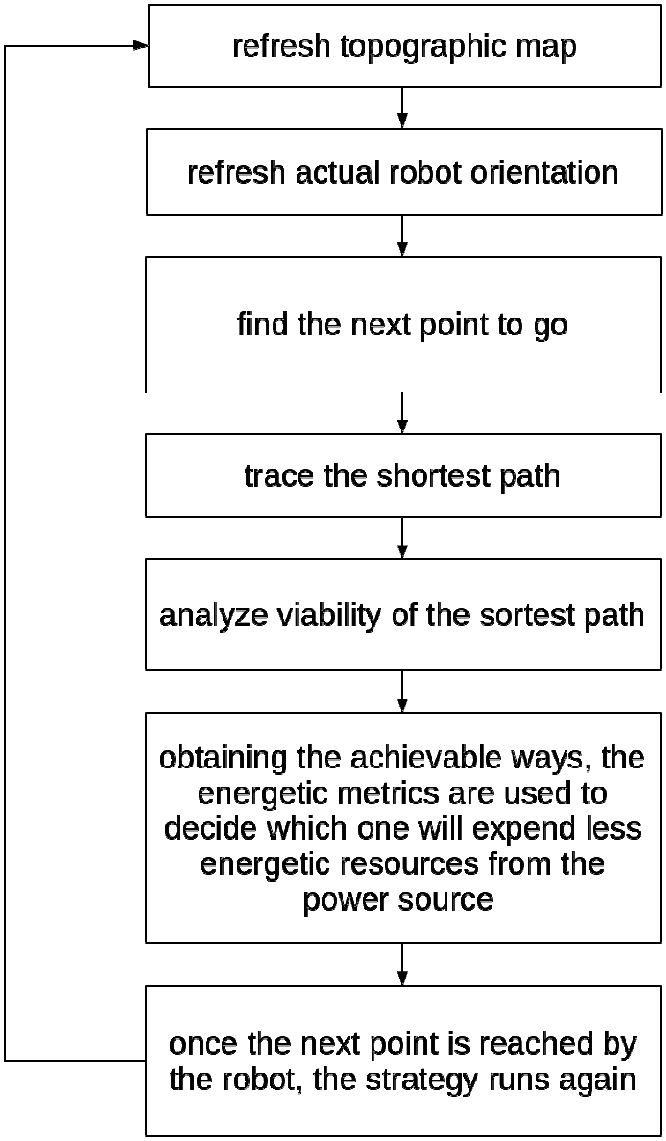

refresh topographic map
refresh actual robot orientation
find the next point to go
trace the shortest path
analyze viability of the sortest path
obtaining the achievable ways, the energetic metrics are used to decide which one will expend less energetic resources from the power source
once the next point is reached by the robot, the strategy runs again

FIG. 21

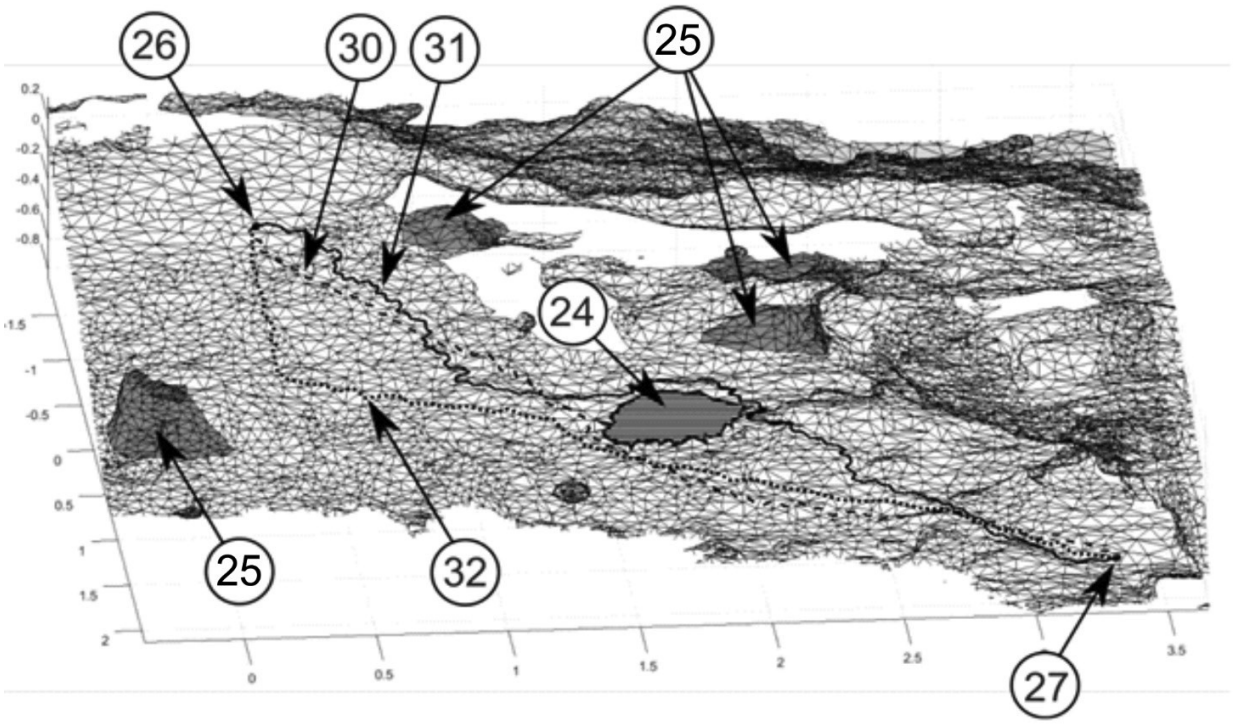


FIG. 22

MULTI-TERRAIN INSPECTION ROBOTIC DEVICE AND METHODS FOR CONFIGURING AND GUIDING THE SAME

FIELD OF THE INVENTION

[0001] The present invention addresses a robotic device capable to inspect confined and risk areas. The robot applies a reconfigurable locomotion module in order to operate in irregular and rough terrains, carrying a mapping unit capable to model the covered terrain as a 3D colored point cloud. The terrain model works as an input of an operating assist module that indicates non-transposable and tipping-over risk areas and suggests adequate locomotion configurations to transpose each obstacle. An autonomous navigation module also uses the terrain model to generate optimal paths considering the traveled distance, tipping-over risk, and energy consumption.

BACKGROUND OF THE INVENTION

[0002] Natural caves are commonly found in iron formation areas. In order to economically explore such areas, an inspection of the caves is environmentally and legally required. Considering this scenario, an inner investigation is crucial to evaluate the cave's relevance and consequently to determine its preservation or exploitation. Speleologist groups are normally gathered to perform those studies. However, natural caves may be hazardous environments due to the presence of venomous animals, noxious gases, bat excrement, risk of roof collapse, among others. The presence of humans in such areas is always related with health hazard. Taking that into account, the use of robotic devices to enter the caves and perform these inspections can be an adequate technical solution.

[0003] From the robotics point of view, natural caves can be extremely challenging. They may present characteristics such as: irregular terrain; closed environments; wireless communication difficulty; non-presence of GPS signal.

A special challenge lays on the topography, which is commonly complex. The terrains are not structured, presenting a mix of flat and rugged areas. Considering such characteristics, the exploring device must have an efficient locomotion system, allying obstacle transposing without tipping-over, energy efficiency and payload capabilities.

[0004] The exploring device must have energy autonomy to entry the cave, perform the inspection in all of inner sectors and return to the operational base. That way, to have a locomotion configuration that is energy consumption friendly, such as wheels, is an effective solution. On the other hand, when compared to legs, wheels are less efficient to transpose obstacles. A locomotion configuration based on legs is more effectively on rough terrain transposing, although it consumes extra energy.

[0005] Another problem related to cave's inspection is the robot's stability. Stability here refers to the capability of the device to maintain itself upward by its own means, without tipping-over. To perform special inspections and 3D mapping, extra sensors must be embedded in the device, adding payload to the system. Usually, the hardware is sensible and should not suffer mechanical impacts, which eventually happens in a rollover event. Regarding that, it is empirical to maintain the device's stability during all the inspection.

[0006] Although it is common to find robotic devices projected to perform environmental inspections, only a few of them are specially designed for natural cave inspection. Some authors claim that robots designed for underground mines inspections fit for the job, but comparing the conditions of this kind of environment with natural caves, the second one requires very different mobility capabilities.

[0007] Specifically, for underground mines inspection, researches at Carnegie Mellon University proposed the development of the Groundhog robot.

It weighs 700kg, measures 1m in height, 1.2m in width and is able to map underground mines in teleoperated or autonomous modes. However, its large dimensions make it not suitable for natural caves inspection. The robotic device can get stuck in a confined area or damage the ambiance fragile structure.

[0008] Still on Carnegie Mellon University, the U.S. Pat. No. 7,069,124 describes a robotic method for void mapping. The authors disclosure two robots that are chosen for a specific mission depending mainly on the void entrance, leaving in the background the terrain structure conditions. This is a plausible approach since underground mines commonly have a structured terrain, rarely presenting obstacles on the way or extreme rough terrains. As the authors claim, the robots can move over flat and half-rugged terrains, being able to overcome some obstacles; however, rougher terrains are non-transposable and the strategy to map those areas is to install the sensors at the device´s robotic arm, and stretch the arm until its workspace frontier.

[0009] The Counter Tunnel Exploitation Robot (CTER), developed by the SPAWAR Systems Center Pacific, is a small robot designed to inspect smuggling tunnels. It has small size and uses a locomotion configuration based on steer tracks. Its body is long and flexible, allowing the device to pass through small spaces and holes in order to access the tunnels. Still, the CTER’s design contemplates the ability to enter in confined areas with access restriction, rather than transposing rough terrain.

[00010] Designed specifically for natural caves inspection, the robot FREESE from SILES, I. and WALKER, I. D. has small dimensions, star-shaped locomotion mechanisms and is assembled on a flexible frame. Those characteristics guarantee great mobility in natural caves; however, it has small payload capability, not allowing it to carry extra sensors or other equipment to perform a full 3D colored mapping of the environment.

In comparison to the mobility solutions considered state of art, the invention proposed in this document solves several locomotion restrictions by providing a solution to quickly change the mobility characteristics of the inspection device, and evaluate the topographical map of the surroundings in order to indicate which locomotion configurations are able to transpose each part of the terrain.

[00011] Regarding devices for 3D mapping, most of commercial 3D laser scanners are stationary equipment set on tripods by specialists to perform sweeps where mapping is demanded. They used to be compound by laser sensors, mirror arrays, HD cameras, GPS and Inertial Measurement Unit (IMU). Thus, when scanning, point clouds and RGB imaging are collected to be associated with and post-processing on software to generate 3D colored maps.

[00012] Despite the equipment, there is a particular solution in 3D laser scanning named Zebedee, patented under AU2016205004 by CSIRO, in Australia. It consists of a 2D LiDAR and a MicroElectroMechanical System (MEMS) IMU mounted on a spring mechanism. As the operator moves through the intended environment, the scanner loosely oscillates about the spring producing a rotation that converts 2D measurements into 3D fields of view. Similar to the other solutions, it is necessary direct human interaction in order to provide results.

OBJECTS OF THE INVENTION

[00013] The object of the invention consists on a robotic device that is capable to inspect and map confined and risk areas, i.e., caves, sewer and dam spillway galleries, and areas with risk of collapse, being able to overcome unstructured terrains and model the surrounding environment through a 3D colored point cloud. The device counts on a reconfigurable locomotion module, which the specific configuration is defined based on the terrain model. The terrain model also provides information to indicate non-transposable and

tipping-over risk areas, and to generate optimal paths considering traveled distance, tipping-over risk, and energy consumption.

BRIEF DESCRIPTION OF THE DRAWINGS

[00014] Figure 1. Orthogonal view of the exploded robotic device frame;

Figure 2. Front view of the robotic device frame;

Figure 3. Side view of the robotic device frame;

Figures 4(A) to (E). Sequence of steps to quickly change the locomotion mechanism;

Figure 5. Detailed view of the arc-shaped leg locomotion mechanism;

Figure 6. Detailed view of the common wheel locomotion mechanism;

Figure 7. Detailed view of the star-shaped wheel locomotion mechanism;

Figure 8. Detailed view of the steer track locomotion mechanism;

Figure 9. Six common wheels locomotion configuration;

Figure 10. Six arc-shaped legs locomotion configuration;

Figure 11. Four common wheels and two arc-shaped legs locomotion configuration;

Figure 12. Four common wheels and two star-shaped wheels locomotion configuration;

Figure 13. Six star-shaped wheels locomotion configuration;

Figure 14. Four common wheels with steer tracks locomotion configuration;

Figure 15. Detailed view of the mapping unit;

Figure 16. Exploded view of the mapping unit's tip;

Figure 17. Mapping unit operational workflow;

Figure 18. A point cloud and generated mesh;

Figure 19. Device's teleoperation scheme;

Figure 20. Topographic map from the cave showing the transposable areas by the actual locomotion configuration, transposable areas by another locomotion configuration and tipping-over risk areas;

Figure 21. Flowchart describing the autonomous navigation module operation;

Figure 22. Demonstration of a topographic map with different paths calculated based on mobility policy.

DETAILED DESCRIPTION OF THE INVENTION

[00015] Natural caves have a very peculiar terrain topography. Depending on the geological formation, they may present from a smooth and structured to a rough and highly non-structured terrain. Furthermore, a single cave commonly has different kinds of terrain, what composes a heterogeneous topography with different requirements for optimal terrain transposing.

[00016] Considering that, the present invention consists on a single robotic device whose locomotion characteristics can be easily and quickly modified according to each kind of terrain presented in the cave. An operating assist module analyzes the terrain’s topography, and based on that evaluate the tipping-over risk areas, suggesting possible locomotion configurations to transpose obstacles along the terrain. Each component and module present in this invention is explained below.

Robotic device’s construction

[00017] The robot frame has a rectangular body shape, as can be seen in FIG. 1, FIG. 2 and FIG. 3. The robot carries a core computer 1, a wireless communication module 2 and two batteries 3. There is also a set of cameras and illumination modules located on the front 4 and back 5 of the robot. The

illumination modules are composed of a white light bright led and infrared lights. The cameras are full HD with wide-angle lens.

[00018] The locomotion mechanisms actuation uses six rotational joints, being three on each side of the body. Each joint axle 6 is actuated by a DC motor 7, possessing a total of six independently motors that can be controlled in position, velocity or torque modes. Each motor is coupled to the axle by a planetary gear 8 and a drive belt 9.

Quick reconfigurable locomotion module

[00019] The proposed robotic device has different locomotion mechanisms that can be quickly replaced. It allows the same device to have different locomotion characteristics and capabilities. FIG. 4 demonstrates how this module works. It is based on a quick release/attach pin to change the locomotion mechanism. The first box shows the locomotion mechanism attached to the joint axle by the pin (FIG. 4.A). To change a specific mechanism, it is necessary to release the pin (FIG. 4.B), release the locomotion mechanism (FIG. 4.C), insert in the axle the desired locomotion mechanism (FIG 4.D) and finally attach the pin (FIG 4.E).

[00020] This module dispenses the use of any additional tool. This is highly desirable as natural caves are commonly found on difficult access regions, and all the hardware must be carried by the operators; considering this, the less weight is necessary to be carried, the better.

[00021] The locomotion mechanisms proposed this invention are: arc-shaped legs 10 (FIG. 5), common wheels 11 (FIG. 6), star-shaped wheels 12 (FIG. 7) and steer track 13 (FIG. 8). The assembly configuration pattern can be homogeneous, using only one kind of locomotion mechanism, or hybrid, using different types together. The assembly pattern can be configured, but not limited to, as: six (6) common wheels 11 (FIG. 9); six (6) arc-shaped legs 10 (FIG. 10);

four (4) common wheels 11 on the corners and two (2) central legs 10 (FIG. 11); four (4) common wheels 11 on the corners and two (2) central star-shaped wheels 12 (FIG. 12); six (6) star-shaped wheels 12 (FIG. 13); four (4) wheels 11 on the corners with the steer tracks 13 coupled on (FIG. 14). Exceptionally, the steer tracks are not attachable directly on the joint axle; instead, they are coupled to two common wheels. In this case, any available locomotion mechanism can be attached to the central axles in order to help on the locomotion.

[00022] The assembly pattern is referred in this document as "locomotion configuration"; consequently, each configuration is a combination of locomotion mechanisms. Each locomotion configuration has its own advantages and limitations. As an example, configurations based on common wheels consume less energy but are less efficient for obstacle transposition. When using legs, the characteristics are most opposite, enabling the device to move over rougher terrains but consuming more energy. In comparison with common wheels and legs, star-shaped wheels present intermediate performance. Steer tracks are most indicated for muddy terrain and present power consumption greater than wheels. In short, each locomotion configuration is indicated for a different kind of terrain, and the decision of which one should be used in the mission is fundamental to optimize the robot’s traveled distance, stability and energy consumption.

Mapping unit

[00023] The mapping unit 14 is illustrated in FIG. 15 and the exploded view of its tip can be seen in FIG. 16. It is installed at the top of the robot to generate 3D colored point cloud datasets representing the surrounding environment, combining 3D geometric data and colored high-resolution images from cave walls. Using a DC servomotor, the mapping unit rotates 360° continuously around the robot collecting and managing data from sensors. For

that, the mapping unit is composed by an ultra-compact computer unit and sensors including a laser scanner - LiDAR 15, RGB cameras with large field-of-views lenses 16, high brightness LED external illuminators 17 and an Inertial Measurement Unit (IMU). All components are water and dust proof with IP Code from 54 to 67. Alternatively, all the components can be installed inside the robot or the mapping unit chassis, which are fully protected against external influences.

[00024] For software development purposes, the mapping process is divided into sub-functionalities: scanning manager, unit rotation, images capturing and points capturing (FIG. 17).

- Scanning Manager

[00025] The scanning manager is responsible for coordinating the process of data collection performed by the LiDAR. After sending a command through the command interface, a state machine execution starts. The first action of the state machine is to check the status of the battery, to evaluate the feasibility of running the shot. Later, the state machine sends a command to connect the LiDAR, the two cameras, the 4 LEDs and the DC servomotor. The execution identifies a zero point of the rotation as the initial reference for the shot. Subsequently, LiDAR point storage starts. The images capturing are done at 08 (eight) rotation points (mapping unit angles: 0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°, 360°). In case of failure, an alert message is displayed, and the user decides when to restart the scanning manager. The user can also stop the scanning at any stage. In the event of successful completion, the components are shut off and the mapping unit is repositioned.

[00026] The proposed system can map the environment in different operation conditions, including the robot stopped or moving. In order to attenuate the vibrations caused during the mapping while the robot is moving,

the mapping unit may be connected to the robotic device by suspensions made of springs and/or airbags.

- Mapping unit rotation

[00027] The mapping unit shall rotate about a vertically disposed axis using straight gears transmission and a DC servomotor. A 360° turn of the mapping unit consists of 4 full turns of the DC servomotor; due to the presence of gears, each complete rotation of the DC servomotor represents a 90° turn of the unit. The DC servomotor encoder indicates the end of the mapping unit rotation after the end of the 4th turn, consequently finishing the data acquisition. For the image storage step, this functionality should send the encoder angle of the DC servomotor and the quadrant at the time of capture.

- Images capturing

[00028] For image capture, two cameras are positioned according to Figure 16, being possible to adjust their positioning. After starting a shot, the images are captured from both cameras simultaneously. This data is stored during the execution of the shot.

- 3D Points capturing

[00029] A LiDAR provides 3D geometric data to generate a surface of points with their respective depths. It will return the points coordinates (X, Y, Z) referring to the coordinate system of the LiDAR. This handle data compiles a file called a point cloud, which is generated at the end of the shot. After this step, calibration is executed to correlate the point cloud with the photos taken by the HD cameras. For each 3D point, a pixel with the corresponding color of the photo collected during shot execution is related, so that a colored 3D point cloud is generated.

[00030] It is possible to generate a 3D mesh from a point cloud. A 3D mesh is a collection of planes, being triangular in the case of the present

invention, which represents a three-dimensional shape. Therefore, the 3D mesh (FIG. 18.A) obtained from the point cloud (FIG. 18.B) represents the topography of the robot surroundings.

Operating assist module

[00031] Tele-operating an inspection robot is generally related to providing the operator a command interface where is possible to send commands to the device, and receive camera and other sensors feedback in order to visualize the robot surrounding environment. In the case of the present invention, the images provided by the two embedded cameras are shown to the operator at the user interface 20, and a joystick 21 is used to send commands to the robot.

[00032] A major disadvantage on teleoperation, when compared to local operation, is the loss of surroundings notion by the operator. Hereby, operating assist modules are desirable as they facilitate the mission execution and allow the operator to focus his attention on high-level tasks. The proposed assist module helps the operator indicating areas where the robot may tip-over, and suggests adequate locomotion configurations capable to transpose each stretch of the terrain.

[00033] The operating assist module uses as input the topographic map 22 of the robot surrounding environment disregarding cave's walls and ceiling 23. Based on that, the assist module estimates the robot pose (position and orientation) and the stability, related to the risk of tipping-over, for each stretch of the known map. For that, different mobility metrics can be employed, including force-angle measure of tip-over stability margin proposed by Evangelos Papadopoulos and Daniel A. Rey, and energy stability margin proposed by Messuri and Klein. The robot stability prediction allows highlighting the tipping-over risk areas 24 of the map that should be avoided

during all operation.

[00034] For the remaining map area, the topography is analyzed to identify regions that are transposable 22 by the locomotion configuration in use. Given the regions classified as non-transposable 25, other available locomotion mechanisms are evaluated to identify the adequate ones for transposing such obstacles. This evaluation is done through online or offline simulations, which analyze the robot mobility performance using all possible locomotion configurations while transposing similar obstacles. If none of the available locomotion mechanisms are able to overcome these regions, they should also be highlight in the map as areas to be avoided during operation.

[00035] The operating assist module uses the information to generate a modified map of the terrain that explicitly indicates to the user the non-transposable or rollover risk areas that should be avoided, and obstacles where another locomotion mechanisms are more indicated for transposition.

[00036] Thus, the present invention also proposes a method for guiding the multi-terrain inspection robotic device. The robotic device is guided according to the following steps: obtaining an updated topographic map of the robot surroundings; estimating a robot pose over numerous map locations, considering all possible locomotion configurations; calculating stability metrics for all of those estimated poses; identifying areas in the map where the robot may tip-over with the locomotion mechanism in use, analyzing the metrics results obtained with other locomotion mechanism and identifying configurations that do not tip-over while transposing such obstacles; and generating a map with the transposable, tipping-over risk and non-transposable areas with the locomotion mechanism in use, also indicating other locomotion configurations to transpose the restricted areas.

Autonomous navigation module

[00037] The operation in autonomous mode is also available. This strategy differs from classical approaches as it takes into consideration the robot locomotion configuration to calculate its stability and obstacle transposition capabilities for an optimal path planning.

[00038] Before the mission begins, the operator indicates to the module which mobility metric should be optimized: shortest path, minimum tipping-over risk or minimum energy consumption. This information corresponds to specific gains that the operator sets, also allowing to establish a trade-off between the different mobility metrics.

[00039] Overcoming high obstacles normally means to take shortcuts and then perform faster inspections. However, covering rough terrains presents more risk for the robot to tip-over, and consumes more energy considering the extra power applied by the motors to transpose obstacles in the environment. That way, traveling through flatter terrains should be safer and energy consumption friendlier.

[00040] The autonomous navigation module requires as input the updated topographic map from the surroundings. Using stability metrics and the information about the locomotion configuration in use, the module withdraws the tipping-over risk and non-transposable areas from the map; that way, the path planning will not consider risk areas.

[00041] Three paths to the target point are calculated - the shortest, the safer according to the risk of tipping-over, and the energy saver. The choice of which one is going to be executed will depend on the mobility policy defined previously.

[00042] This strategy (FIG. 21) is implemented as follow: the robot is located in the start point 26 and should move the next goal point 27, which can be directly informed by the operator or obtained via exploring algorithms such

as cellular decomposition for complete coverage path planning. The non-transposable obstacles 25 by the locomotion configuration in use and the tipping-over risk areas 24 are not considered by the path planning algorithm. The shortest 30, safest 31 and energy friendliest 32 paths are calculated; the executed path is defined according to the mobility policy defined previously.

[00043] After exploring all possible regions transposable by the actual locomotion configuration, the robot returns to the base and indicates to the operator an alternative locomotion mechanism capable to access the non-inspected areas. Using the recommended locomotion configuration, the robot is able to re-enter the mission area, going directly to the non-covered areas in order to complete the 3D map of the inspected environment.

[00044] Thus, the present invention further provides a method for guiding the multi-terrain inspection robotic device. The robotic device is guided according to the following steps: obtaining an updated topographic map of the robot surroundings; obtaining gain values representing the mobility policy, which indicates which metric to optimize: shortest path, minimum tipping-over risk or minimum energy consumption; obtaining a robotic device's next goal point, which can be directly informed or obtained via exploring algorithms such as cellular decomposition for complete coverage path planning to inspect the entire environment; removing from the topographic map areas with tipping-over risk and non-transposable areas by the locomotion mechanism in use; tracing, considering the modified map generated, the shortest path to the target point using path planning algorithms such as A* or D*; tracing, considering the modified map, the path with less tipping-over risk to the target point using path planning algorithms based on stability metrics, such as force-angle measure of tip-over stability margin or energy based stability margin; tracing, considering the modified map, the optimal energy consumption path to the target point using

path planning algorithms based on the energy consumption of the robot motors; choosing which one of the obtained paths is going to be used depending on mobility policy defined for the autonomous navigation module; commanding the robotic device to reach the goal point.

[00045] While various example embodiments have been described above, it should be understood that they have been presented by way of example, and not limitation. It will be apparent to persons skilled in the relevant art(s) that various changes in form and detail can be made therein.

CLAIMS

1. A multi-terrain inspection robotic device comprising:

a robot frame, comprising a plurality of cameras with an illumination system (4, 5);

a mapping unit (14) attached to the robot frame, comprising a plurality of sensors (15);

a reconfigurable locomotion module comprising a plurality of locomotion mechanisms arranged according to selectable assembly patterns;

an operating assist module for receiving data from said sensors and cameras and identifies the adequate assembly pattern of locomotion mechanisms; and

an autonomous navigation module for determining an optimal path planning for the robotic device,

wherein the autonomous navigation module is further configured to indicate an alternative assembly pattern of locomotion mechanisms.

2. The multi-terrain inspection robotic device of claim 1, wherein the robot frame has a rectangular shape with six rotational joints (6), being three for each side.

3. The multi-terrain inspection robotic device of claim 2, wherein each joint (6) has its own independent controllable motor.

4. The multi-terrain inspection robotic device of claim 1, wherein the locomotion mechanisms of the reconfigurable locomotion module are selected from the group composed by the following: common wheels (11);

star-shaped wheels (12); arc-shaped legs (10); and steer tracks (13).

5. The multi-terrain inspection robotic device of claim 4, wherein the locomotion mechanisms of the reconfigurable locomotion module are promptly and unitarily replaced using a quick release pin.

6. The multi-terrain inspection robotic device of claim 4, wherein the reconfigurable locomotion module enables the assembly pattern to be homogeneous, using only one type of locomotion mechanism, or heterogeneous, using different types of the available locomotion mechanisms together, the assembly pattern can be configured, but no limited only to, as: 6 common wheels (11); 6 star-shaped wheels (12); 6 arc-shaped legs (10); 4 common wheels (11) on the corners and 2 central legs (10); 4 common wheels (11) on the corners and 2 central star-shaped wheels (12); 4 common wheels (11) on the corners with the steer tracks (13) coupled on.

7. The multi-terrain inspection robotic device of claim 1, wherein the mapping unit (14) comprises as sensors a 3D laser scanner (15), full HD color cameras (16), high brightness LED external illuminators (17) and an Inertial Measurement Unit.

8. The multi-terrain inspection robotic device of claim 1, wherein the mapping unit (14) has a laser scanner (15) in a vertical position and its tip has a fully 360° rotational mechanism, which enables the sensor to scan all of the robot surroundings.

9. The multi-terrain inspection robotic device of claim 7, wherein the raw data provided by the mapping unit sensors includes 3D point cloud and colored images.

10. The multi-terrain inspection robotic device of claim 9, wherein the point cloud and colored images data are fused to generate colored 3D point cloud of the inspected area.

11. The multi-terrain inspection robotic device of claim 8, wherein the point cloud is used to generate a 3D mesh of the area, which is further used as a topographic map of the robotic device surroundings.

12. The multi-terrain inspection robotic device of claim 1, wherein the robotic device is specifically designed for operating in mining activities.

13. The multi-terrain inspection robotic device of claim 1, wherein the robotic device is specifically designed for operating in speleology activities.

14. The multi-terrain inspection robotic device of claim 16, wherein the robotic device is specifically designed for cave inspections in speleology activities.

15. The multi-terrain inspection robotic device of claim 1, wherein the robotic device is specifically designed for operating in confined areas.

16. The multi-terrain inspection robotic device of claim 18, wherein the robotic device is specifically designed for operating in sewer and dam spillway galleries;

17. The multi-terrain inspection robotic device of claim 1, wherein the robotic device is specifically designed for operating in areas with risk of collapse.

18. Method for guiding the multi-terrain inspection robotic device of claim 1, the method comprising the following steps:

obtaining an updated topographic map of the robot surroundings;

estimating a robot pose over numerous map locations, considering all possible locomotion configurations;

calculating stability metrics for all of those estimated poses;

identifying areas in the map where the robot may tip-over with the locomotion mechanism in use, analyzing the metrics results obtained with other locomotion mechanism and identifying configurations that do not tip-over while transposing such obstacles; and

generating a map with the transposable, tipping-over risk and non-transposable areas with the locomotion mechanism in use, also indicating other locomotion configurations to transpose the restricted areas.

19. The method of claim 18, wherein the tipping-over risk estimation is based on stability metrics, the metrics being applied on the topographic map of the surroundings, considering the robot's geometry according to a given locomotion configuration and estimated poses and wherein the stability metrics to be considered are the force-angle measure of tip-over stability margin and energy stability margin.

20. Method for guiding the multi-terrain inspection robotic device of claim 1, the method comprising the following steps:

obtaining an updated topographic map of the robot surroundings;

obtaining gain values representing the mobility policy, which indicates which metric to optimize: shortest path, minimum tipping-over risk or

minimum energy consumption;

obtaining a robotic device's next goal point, which is either directly informed or obtained via exploring algorithms such as cellular decomposition for complete coverage path planning to inspect the entire environment;

removing from the topographic map areas with tipping-over risk and non-transposable areas by the locomotion mechanism in use;

tracing, considering the modified map generated, the shortest path to the target point using path planning algorithms;

tracing, considering the modified map, the path with less tipping-over risk to the target point using path planning algorithms based on stability metrics, such as force-angle measure of tip-over stability margin or energy based stability margin;

tracing, considering the modified map, the optimal energy consumption path to the target point using path planning algorithms based on the energy consumption of the robot motors;

choosing which one of the obtained paths is going to be used depending on mobility policy defined for the autonomous navigation system;

commanding the robotic device to reach the goal point.

ABSTRACT

MULTI-TERRAIN INSPECTION ROBOTIC DEVICE AND METHOD FOR GUIDING THE SAME

This disclosure presents a robotic device for multi-terrain inspection, composed by a robot body, a quick reconfigurable locomotion module and a mapping unit capable to model the inspected environment through a 3D colored point cloud. The robot has different locomotion mechanisms that can be quickly replaced, thereby changing the robot mobility characteristics. The device is controlled through teleoperation or autonomously. When in teleoperated mode, an operating assist module provides relevant locomotion information to the operator including a map that shows areas where the robot may not transpose or tip-over. This module also suggests to the operator other locomotion configurations to overcome obstacles presented in the map. When in autonomous mode, the navigation module provides a strategy to explore unknown environments and trace optimal locomotion path considering the traveled distance, tipping-over risk and energy consumption. Regarding the invention characteristics described above, the main objective is to perform inspections of confined and risk areas, i.e., caves, sewer and dam spillway galleries, and areas with risk of collapse.