COPPE
Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia
UFRJ

TRANSPOSIÇÃO AUTÔNOMA DE ESCADAS PARA ROBÔS MÓVEIS COM BRAÇOS ARTICULADOS SOBRE ESTEIRAS

Thales Henriques da Silva

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Fernando Cesar Lizarralde

Rio de Janeiro
Maio de 2022

TRANSPOSIÇÃO AUTÔNOMA DE ESCADAS PARA ROBÔS MÓVEIS COM BRAÇOS ARTICULADOS SOBRE ESTEIRAS

Thales Henriques da Silva

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Orientador: Fernando Cesar Lizarralde

Aprovada por: Prof. Fernando Cesar Lizarralde
Prof. Ramon Romankevicius Costa
Prof. Gustavo Medeiros Freitas
Prof. Guilherme Augusto Silva Pereira

RIO DE JANEIRO, RJ – BRASIL
MAIO DE 2022

Silva, Thales Henriques da

Transposição autônoma de escadas para robôs móveis com braços articulados sobre esteiras/Thales Henriques da Silva. – Rio de Janeiro: UFRJ/COPPE, 2022.

XII, 68 p.: il.; 29,7cm.

Orientador: Fernando Cesar Lizarralde

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2022.

Referências Bibliográficas: p. 65 – 68.

1. Subida autônoma de escadas. 2. Robótica autônoma. 3. Robôs sobre esteiras. 4. Controle cinemático. I. Lizarralde, Fernando Cesar. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

TRANSPOSIÇÃO AUTÔNOMA DE ESCADAS PARA ROBÔS MÓVEIS COM BRAÇOS ARTICULADOS SOBRE ESTEIRAS

Thales Henriques da Silva

Maio/2022

Orientador: Fernando Cesar Lizarralde

Programa: Engenharia Elétrica

Esta dissertação apresenta uma nova abordagem para o problema de subida e descida de escadas para robôs equipados com braços articulados sobre esteiras. Considera-se um robô dotado de um meio principal de locomoção, como esteiras ou rodas, junto à braços que podem ser usados para estender sua mobilidade quando necessário. A ideia principal consiste em interpretar o robô como um manipulador planar com uma restrição de posição. Para o modelo proposto, uma realimentação de estados com propriedades de estabilidade e convergência é então desenvolvida para o gerenciamento autônomo dos braços durante o processo de transposição. O controlador age adequando as esteiras aos planos imediatamente abaixo sempre que possível, fazendo com que o robô realize manobras de que controlam a orientação, prevenindo movimentos bruscos, melhorando a tração e evitando colisões entre a estrutura do robô e os degraus. O método apresentado é uma nova forma de interpretar o problema. O esquema de controle proposto é validado em um robô real e resultados experimentais são apresentados.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

STAIR CLIMBING AND ARMS MANAGEMENT CONTROL FOR VEHICLES WITH ARTICULATED TRACKED ARMS

Thales Henriques da Silva

May/2022

Advisor: Fernando Cesar Lizarralde

Department: Electrical Engineering

This work presents a new approach to the problem of ascending and descending stairs for robots equipped with articulated arms on tracks. A robot is considered to have a main locomotion system, such as tracks or wheels, along with arms that can be used to extend its mobility when necessary. The main idea is to interpret the robot as a planar manipulator with a position constraint. For the proposed model, a state feedback control law with stability and convergence properties is then developed for the autonomous management of the arms during the transposition process. The controller adapts the tracks to the planes immediately below whenever possible, making the robot perform maneuvers that control the orientation, preventing sudden movements, improving traction and avoiding collisions between the robot structure and the steps. The method presented is a new way of interpreting the problem. The proposed control scheme is validated in a real robot and experimental results are presented.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O emprego de robôs em ambientes urbanos ocorre por motivos variados, que vão desde o simples auxílio ao operador, a cenários mais complexos, onde deseja-se removê-lo do ambiente em função do risco à vida. Uma vez que estes ambientes são em sua maioria projetados para humanos, robôs vão inevitavelmente continuar sendo desafiados por obstáculos que são queridos por nós e, ao mesmo tempo, terríveis para eles. Neste grupo de obstáculos, um é especial: as escadas. Basicamente omnipresentes, elas são o meio mais pragmático de conectar duas superfícies em alturas diferentes. Uma vez que a sociedade moderna enxerga construções verticais como o meio para lidar com o problema da falta de espaço, num mundo com cada vez mais pessoas e menos espaço, espera-se que elas continuem presentes por muito tempo. Como se pode esperar, algoritmos para subida de escada tem sido propostos para robôs terrestres de diversas configurações.

Robôs terrestres podem ser agrupados em três classes principais a depender do seu sistema de locomoção: pernas, esteiras e rodas [4]. Robôs sobre pernas são conhecidos por sua grande adaptabilidade à terrenos não estruturados, em muitos casos superando robôs sobre esteiras ou rodas [19]. Apesar de serem conhecidos por seu alto grau de acoplamento e problemas de controlabilidade, alguns artigos tem sido publicados sobre o assunto. Em [1] uma análise do desempenho de robôs sobre pernas no *DARPA Robotics Challenge* é apresentada. Dada sua natureza fortemente não-linear e centro de massa elevado, estes robôs dependem de controle em malha fechada apenas para se manterem de pé. Durante o desafio, alguns participantes tiveram dificuldades ao se deslocar por terrenos irregulares provocando um número elevado de quedas. Outra configuração de robôs sobre pernas que tem despertado o interesse da comunidade acadêmica por seus ótimos resultados são os quadrúpedes, como AiDIN-VI [23], ANYmal [14] e Mini-Cheetah [22]. Apesar de sua grande mobilidade e sucesso, estes robôs são relativamente lentos, consomem uma grande quantidade de energia, além de não serem ideais para transpor longas distâncias [4].

Como forma de agrupar as melhores qualidades de cada um dos meios de locomoção em um único robô, configurações híbridas são adotadas, combinando pernas, esteiras e ro-

das. Grande parte da literatura sobre subida de escadas emprega robôs híbridos que usam esteiras como meio de locomoção principal, dotados de braços articulados (que podem ser interpretados como pernas). Em [17] estes robôs são referidos pela sigla AATV, acrônimo para Actively Articulated Tracked Vehicle. Conhecidos por sua robustez, eficiência energética e relativa rapidez (quando equipados com rodas), estes robôs podem usar os braços para superar obstáculos e terrenos difíceis. Apesar disso, o alto número de graus de liberdade envolvidos junto a ausência de compreensão do cenário de operação elevam demasiadamente a carga de trabalho do operador, especialmente durante a transição de obstáculos, como escadas.

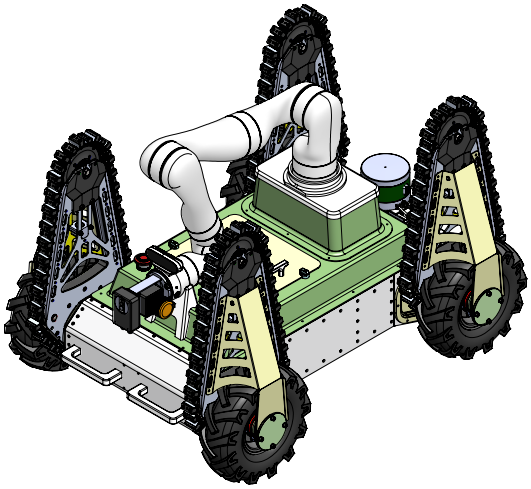


Figura 1.1: O robô móvel utilizado nos resultados experimentais chamado de Rosi. O sistema primário de locomoção são rodas, contudo, os braços sobre esteiras podem ser usados para transpôr obstáculos e terrenos difíceis.

Para que um AATV como Rosi, na figura 1.1 [33] seja capaz de superar uma escada, inicialmente seus braços devem estar posicionados de forma tal que sejam capazes de puxar todo o corpo do robô. Ao avançar em direção à escada, os braços entram em contato com a borda do primeiro degrau. Deste momento em diante, o robô torna-se uma cadeia cinemática aberta e se estabelece uma relação não-linear entre a posição das juntas e sua pose. Um pequeno avanço em direção à escada ou leve movimento dos braços, inevitavelmente causa uma um desvio das configurações ideais para subida, devendo estas serem corrigidas à cada instante. Em outras palavras, pode-se dizer que uma malha fechada de controle faz-se necessária. Muito embora a literatura sobre o tema para AATVs seja vasta e rica, com exceção de [17], é fato que nenhum outro trabalho trata propriamente do controle dos braços ou tenta modelar a etapa de transição entre um andar e a escada. Parece ser uma parte ausente na literatura. De forma geral, na literatura apresentada mais afrente no texto, os autores fazem uso extensivo de métodos heurísticos e empíricos, controlando os braços apenas posicionando-os de forma fixa baseado em eventos discretos.

1.1 Trabalhos relacionados

Embora as abordagens para o problema sejam diversas, as soluções convergem em alguns pontos, formando o que parecem ser consensos. Por ordem cronológica, a subida de escadas é abordada em [34] para um robô sobre esteiras chamado Urbie, apresentado na figura 1.2. Um sonar, uma câmera monocular e dois acelerômetros são empregados

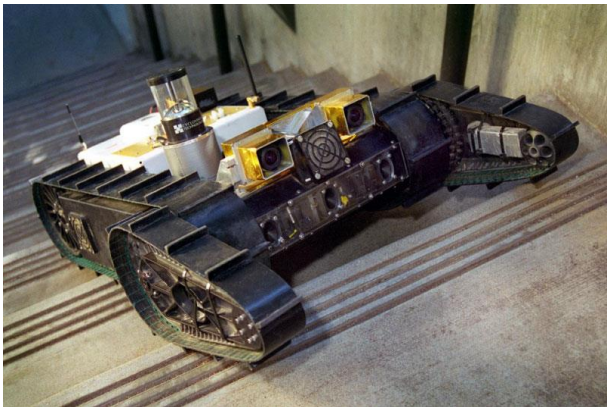


Figura 1.2: O robô Urbie em operação. Note o conjunto de sensores à frente.

para obtenção de estimativas da orientação. Os sensores são usados separadamente sob a hipótese de que cada um fornece boas estimativas sob determinadas circunstancias. Uma lei de controle por realimentação de estados é projetada para guiar o robô escada acima e uma estratégia é implementada para escolher qual estimativas deve ser utilizada no cálculo do erro. O método dos autores para obtenção de uma estimativa da orientação é questionável. Individualmente nenhum dos sensores oferece medidas confiáveis para a situação empregada. Além disso, esteiras e as bordas dos degraus tem uma interação demasiadamente brusca. Escorregamento e pequenas colisões funcionam como uma fonte de vibração prejudicando as leituras dos sensores. A decisão dos autores de não fundir as informações de diferentes sensores para obtenção de melhores estimativas levou à pobreza no sensoreamento. Os autores negligenciam o controle dos braços nas etapas de transição entre a escada e os andares, feito por meio de posições fixas empiricamente determinadas.

Em [29] uma fusão sensorial por meio de um Extended Kalman Filter é apresentada para o robô da figura 1.3, equipado com giroscópios e uma câmera. A velocidade angular dos giroscópios é integrada para obtenção de uma estimativa da orientação a priori. Posteriormente, imagens da câmera são utilizadas para medição de projeções das bordas dos degraus e então a estimativa da orientação é atualizada. A identificação das bordas permite que além da orientação, a distância entre o robô e o centro da escada sejam conhecidas. Com essas duas estimativas os autores propuseram uma lei de controle em malha fechada para guia-lo ao fim da escada centrado sobre a mesma. Comparado a [34], o sensoreamento foi aprimorado.

Em [27] o robô chamado Silver na figura 1.3 é equipado com uma câmera estéreo e

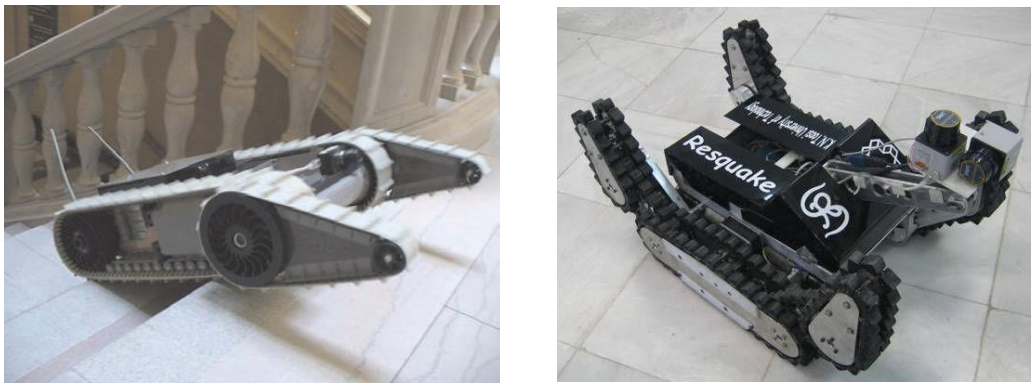


Figura 1.3: O robô similar ao Urbie usado em [29] à esquerda, equipado com uma pequena câmera monocular visível na foto. Na direita, o robô *Silver* usado em [27], com dois laser range finders posicionados na horizontal e vertical.

dois scanners à laser. Os lasers são montados complementarmente de modo a cobrir os planos vertical e horizontal à frente do robô, afim de alimentar um algoritmo para extração de linhas em busca de padrões similares aos de uma escada. Ao encontrar uma escada, o robô se move para uma postura apropriada, definida pelos autores como com uma orientação alinhada ao plano do primeiro degrau e posição próxima à sua largura média. Uma vez sobre a escada, um controlador proporcional é implementado para correção do rumo junto à um controlador fuzzy que atua na velocidade de avanço em função do erro de rumo. Apesar do robô empregado possuir mais um par de braços articulados, conferindo mais graus de liberdade, os autores não apresentam proposta acerca do gerenciamento contínuo dos braços, controlando-os de forma discreta com posições predefinidas.

Uma abordagem diferente é apresentada em [31], onde os autores propõem uma estratégia usando sensores LIDAR para mapear o terreno, junto à um controlador responsável por posicionar os braços de maneira adequada. Em seu trabalho, os autores derivam algumas diretrizes importantes com base na experiência de operadores qualificados operando manualmente o robô em competições de resgate. Dentre outros fatores, eles observaram que para que o robô seja capaz de percorrer suavemente um trecho, sua orientação deve ser mantida de acordo com a inclinação do terreno. Além disso, para permitir uma boa locomoção, as esteiras devem estar em contato com o solo tanto quanto possível. Entretanto, a estratégia de controle proposta pelos autores não se baseia no estudo cinemática diferencial, sendo um tanto quanto inventivo. Além disso, um grande número de sensores são usados: três LIDARs, com dois montados nas laterais e um montado à frente do robô.

Uma abordagem baseada na análise da cinemática diferencial e controle de um AATV é apresentada em [17]. Considerando um modelo conhecido do terreno, os autores propõem o planejamento de uma trajetória livre de colisões, usando todo o potencial dos braços articulados. Um controle para rastreamento de trajetória é usado para atuar de forma autônoma tanto nas esteiras quanto nos braços, de modo que o veículo siga uma dada trajetória 3D desejada. A lei de controle proposta garante a estabilidade do erro de

trajetória e sua convergência para zero. Contudo, a dependência de conhecimento prévio do terreno, nem sempre disponível, e ainda, de estimativas da posição são pontos fracos. Sensores de pose tendem a ser problemáticos e erráticos, sobretudo em ambientes internos, com baixa luminosidade e features de baixa qualidade.

Numa visão mais pragmática, em [13] os autores argumentam que uma escada, de forma geral, são terrenos bem estruturados de forma que métodos mais sofisticados não são necessários. O trabalho apresenta uma estratégia baseada somente no conhecimento da velocidade das esteiras e orientação do robô, adquirida por meio de uma IMU. A escada é modelada em função de sua distância, inclinação e número de degraus. Os pontos críticos são listados: deslizamento na subida, tombamento na navegação e impacto na chegada ao andar superior. Uma estratégia de movimentação inventiva é adotada na tentativa de solucionar tais problemas. Novamente os autores propõe um controlador proporcional para correção do erro de rumo aliado à uma estrategia de posicionamento dos braços em posições fixas, baseada em eventos discretos. O trabalho falha em apresentar novas propostas tanto para o rumo quanto para o gerenciamento dos braços.

De forma geral, a maior parte dos trabalhos presentes na literatura aproximam os AATVs por um robô de tração diferencial, negligenciando derrapagem e outros efeitos, uma vez que um simples controlador proporcional tem se mostrado eficaz em guiar o robô ao fim da escada. Entretanto, durante a subida os efeitos de derrapagem e perda de pontos de contato são facilmente perceptíveis. Um trabalho focado em cadeiras de rodas, [10] faz uma contribuição interessante. Ao invés de modelar o sistema, os autores pressupõem uma planta com dinâmica linear e usam uma técnica para identificação de parâmetros, posteriormente controlando o rumo com um LQR. Junto à identificação do sistema e controle, os autores também propõe uma técnica para identificação de escadas baseada na utilização de filtros sucessivos para extração de bordas e linhas, respectivamente, usando uma câmera RGB-D.

Para interagir com o ambiente de forma segura, os robôs dependem fortemente de modelos confiáveis dos objetos contidos nele. Se esses modelos não estão disponíveis a priori, o robô precisa identificá-los durante sua missão. Desta forma, com exceção de [17] e [31], onde o modelo do terreno não leva em consideração uma escada propriamente, todos os trabalhos apresentados até o momento carregam consigo um problema secundário e essencial: identificar escadas. As abordagens consistem em processar dados vindos de câmeras e lasers com objetivo de identificar features - bordas, linhas e planos na maior parte. Estas features são usadas para localizar geograficamente a escada, e, por vezes, melhorar estimativas de estados como em [29]. Entretanto, o nível de detalhamento varia conforme as premissas das abordagens implementadas. Por vezes, bastando a extração de algumas linhas, como em [10], até um modelo bastante rico do terreno como em [17].

Os autores propõem em [12] uma técnica probabilística para identificar superfícies de suporte que mais tarde são usadas para caracterizar diferentes objetos, dentre os quais,

escadas. Entretanto, o método rendeu uma precisão relativamente baixa. Em [30] a comparação de dois métodos de detecção de escadas é apresentada. A primeira é baseada em uma técnica chamada scan-line grouping (SGL) [6] que detecta as superfícies planas de objetos poliédricos arbitrários em imagens de profundidade. O segundo usa o método two-point based sampling [21] para determinar as direções principais de uma nuvem de pontos conjunto de dados e procura por planos ao longo dessas direções. Embora os resultados obtidos sejam satisfatórios, assume-se que seu conjunto de dados é preenchido principalmente pela escada, como apontado pelo autor.

Em [9], a abordagem proposta consiste em usar Transformada de Hough [2] para extrair padrões de linha 2D embutidos nos dados 3D obtidos. Uma estrutura de dados chamada octree é usada para armazenar e reduzir a resolução os dados de nuvem de pontos 3D obtidos pelos sensores, reduzindo o processamento computacional necessário. Um filtro é aplicado para remover paredes e outros objetos cujas dimensões não sejam compatíveis com as de uma escada típica.

Mais recentemente em [32] uma estratégia usando Deep Learning é apresentada. Imagens de câmera monocular são usadas junto à uma rede neural convolucional treinada para detectar padrões de escada, mostrando bons resultados. Ao detectar-se uma escada, a Transformada de Hough é utilizada na detecção de linhas junto a outros métodos para mitigação de ruído. As linhas são usadas para corrigir o rumo do robô.

Na literatura apresentada sobre o tema, pode-se notar que há um consenso implícito entre os autores sobre quais problemas devem ser resolvidos. Se não há um conhecimento prévio do terreno, o algoritmo para subida de escadas necessita, antes de qualquer coisa, de um modelo da escada a ser superada. Posteriormente, há um consenso entre os autores sobre a importância de algumas variáveis que necessitam ser controladas: (1) o ângulo de rumo e orientação de maneira geral e (2) a distância das bordas. Já sobre a detecção e modelagem de escadas, de forma geral as abordagens consistem na identificação de features como retas, bordas e planos por meio de técnicas de processamento de imagem e filtragem probabilística em conjuntos de dados 2D ou 3D.

1.2 Motivação

Devido a variabilidade dos terrenos e obstáculos em ambientes urbanos, grande parte dos estudos são direcionados aos robôs do tipo AATV [17, 25, 26]. Como visto anteriormente, não é incomum que robôs desta classe estejam equipados com braços dispostos em pares: um par à frente e outro na traseira. Estes braços de forma geral estão conectados a juntas rotacionais ativas e proporcionam uma maior flexibilidade à classe, uma vez que possibilitam a superação de obstáculos mais complicados, uma melhor adequação ao terreno e um controle local da orientação do corpo.

Robôs sobre esteiras são por via de regra robustos, sendo muitas vezes empregados

em aplicações militares [5]. Com uma grande área de contato entre o solo e a esteira, pode-se empregar torques formidáveis uma vez garantida a conversão da potência em movimento. Entretanto, esteiras não são adequadas para percorrer longas distâncias e possuem uma baixa eficiência energética. Assim, robôs sobre esteiras ficam restritos à aplicações locais, uma vez que para cobrir grandes áreas é pouco prático para operador e requer um longo período de tempo de operação com autonomia limitada das baterias.

Figura 1.4: Rosi fazendo inspeção em uma correia transportadora no porto de Itaguaí.

O robô empregado nesta dissertação visa solucionar as limitações da locomoção sobre esteiras ao empregar rodas como seu meio de tração principal. Na figura 1.4, Rosi, como é chamado o robô em questão, foi projetado especificamente para emular capacidades de um operador humano: transpôr grandes distâncias em tempos razoáveis e com mesmo alcance operacional. Rosi é um AATV capaz de alternar o meio de locomoção de acordo com a tarefa. Deslocando-se em terreno minimamente regular, os braços são mantidos erguidos com esteiras desacopladas, garantindo uma velocidade e eficiência invejáveis. Ao encontrar-se sobre um terreno difícil, ambos os pares se adequam ao solo com esteiras acopladas e tracionando, perdendo-se velocidade, mas aumentando a tração e torque disponíveis.

Embora a literatura corrente para subida autônoma de escadas em AATVs seja vasta, em sua maioria costuma negligenciar as etapas de transição entre os andares e as escadas, fazendo uso de métodos um tanto inventivos e empíricos, sem qualquer tipo de modelo matemático. Além disso, ela aborda robôs cujas próprias esteiras são seu principal meio de deslocamento. Entenda-se: o contato entre o fundo do robô e o piso é desejado, caracterizando a situação ideal em que as esteiras principais estarão tracionando. Note na figura 1.4 que Rosi não possui esteiras principais ao fundo. Apenas rodas nas extremidades. Portanto, o dito fundo consiste apenas no alumínio estrutural, de forma que a colisão acidental seria ainda mais danosa. Para estes robôs, subir escadas de forma manual não é trivial para o operador humano, visto que vários compromissos devem ser ponderados: evitar a colisão do fundo, maximizar a tração das esteiras e manter o rumo correto mesmo

em situação de escorregamento. Junta-se à isso o fato de que sobre esteiras o sistema torna-se fortemente acoplado, e um pequeno movimento individual de um braço, além de alterar a orientação do robô, também causa profunda mudança na distribuição de forças de contato entre as esteiras restantes e o chão, resultando em problemas de tração.

1.2.1 Rosi: um robô para inspeção de correias transportadoras

Rosi, apresentada na figura 1.4, é uma plataforma multipropósito para ambientes industriais cujo objetivo principal é a inspeção de correias transportadoras de minério. O robô foi desenvolvido em uma parceria do Instituto Tecnológico da Vale com o Grupo de Sistemas de Controle em Automação e Robótica da Universidade Federal do Rio de Janeiro.

O robô é uma solução teleoperada com capacidades semi-autônomas, proposta para o trabalho de inspeção que normalmente é executado de forma manual por um inspetor, *in loco*, ao longo de quilômetros de correias. As exigências do projeto quanto ao tempo de operação, velocidade de cruzeiro, distância máxima e volume de trabalho do conjunto plataforma e manipulador foram definidas para equiparar a capacidade de um operador humano. Para cumprir as exigências de projeto, um modelo híbrido sobre rodas e esteiras foi desenvolvido, agregando robustez e velocidade ao conjunto. Esta seção apresenta os principais conceitos do projeto no que diz respeito às suas características mecânicas, hardware, software e os sensores embarcados.

Características gerais

Nas figura 1.5, note as diferentes vistas do robô. À esquerda, a distância entre a linha média de pneus e esteiras em lados opostos é de $b_{w_w} = 570\,mm$ e $b_{w_t} = 454\,mm$ respectivamente. À direita, a distância de entre-eixos de $b_l = 630\,mm$. A massa total da configuração apresentada na figura (com manipulador) é de $79\,kg$.

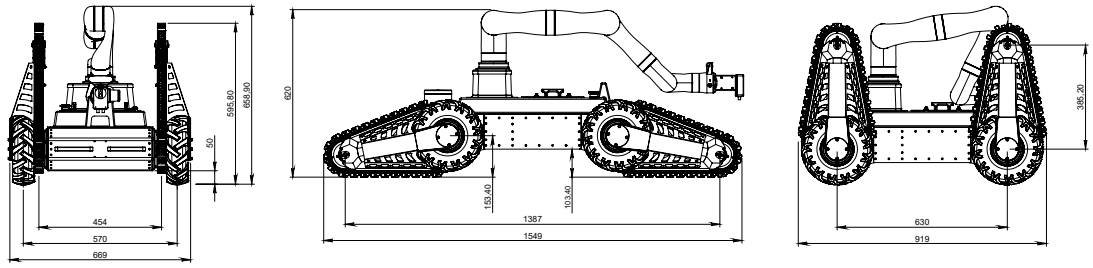


Figura 1.5: Dimensões da Rosi em diferentes vistas.

Rosi possui quatro módulos de tração independentes. Apresentados na figura 1.6, cada módulo consiste de (1) um conjunto roda e pneu de raio $r_w = 127\,mm$, (2) um braço sobre esteira com engrenagem motriz de raio $r_t = 68.3\,mm$, (3) um par de controladores Maxon Epos 4 50/8 e (4) um par de motores Maxon EC 4 pole 200W com torque nominal

de 95.6 $mN.m$ e rotação máxima de 16700 rpm. Dos dois motores, um é para tração e está acoplado à uma redução com relação total de $201.36 : 1$ e outro tem a função de rotacionar o respectivo braço e está acoplado à uma redução de relação total de $1056 : 1$. O conjunto de redução de ambos os motores do módulo de tração é composto por dois estágios de redução, sendo o primeiro Maxon GP 32C com uma eficiência de 70% e o segundo estágio, feito sob medida, com eficiência de 98%. Desta forma, a eficiência total do conjunto de redução é de 68.6%.

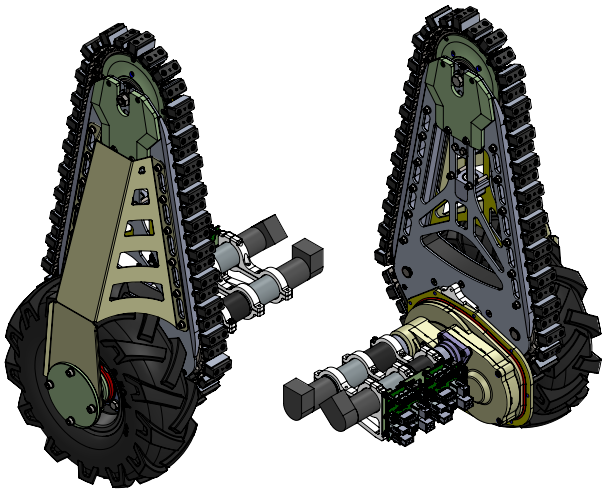


Figura 1.6: Motores, controladores, roda e esteira. Componentes do módulo de tração.

Rosi tem as rodas como seu meio de tração principal. Utilizadas para terrenos regulares, no limite dos motores permitem uma velocidade máxima de 1.1 m/s, similar à um inspetor humano em uma caminhada rápida, um torque máximo de 13.20 $N.m$ no eixo da roda e uma força máxima de 104 N no ponto de contato da roda com o chão. Sobre terrenos difíceis ou no caso de superação de obstáculos, os braços são baixados, levantando as rodas do chão. Este modo de operação garante uma melhor aderência e uma força de 193.26 N distribuídos pela área de contato da esteira com o chão, mas sob a pena de uma velocidade máxima um pouco menor, de 0.6 m/s .

Hardware e software

O robô é equipado com um computador embarcado do tipo PCe/104 com processador Intel Core I7 de quarta geração, 8 GB de memória RAM e um ssd mSATA com Ubuntu 18.04 instalado. O computador se comunica com os controladores de motores via protocolo *CANopen*, que por sua vez viabilizam o acionamento dos motores e acesso à suas variáveis de funcionamento. Todo conjunto é alimentado por até seis baterias Bren-Tronics 24V 9.9Ah, totalizando até 60Ah de carga, se necessário.

A teleoperação é feita por meio de um computador base. O computador base e o robô estão conectados à uma rede Wi-Fi 802.11n. Um ponto de acesso Ubiquiti Rocket

M5 conectado à uma antena Ubiquiti Amo-5g10 se encontram embarcados no robô. Já no computador base, um ponto de acesso da mesma marca está conectado a uma antena Ubiquiti Amo-5g13 proporcionando uma longo alcance sob visada direta entre as antenas da base e robô.

O software do robô utiliza ROS *Melodic Morenia* [1]. Acrônimo para *Robot Operating System*, ROS consiste em um framework com ferramentas e bibliotecas para desenvolvimento de aplicações em robótica. Em ROS, os processos são chamados de *nodes*. Diferentes *nodes* comunicam-se escrevendo e lendo mensagens em tópicos. As mensagens são trocadas à nível da camada de transporte, por meio dos protocolos de comunicação chamados TCPROS e UDPROS, gerenciadas por uma figura chamada *rosmaster*.

Os sensores embarcados são vários: um scanner à laser modelo Velodyne VLP-16, uma IMU/AHRS modelo Advanced Navigation Motus, uma câmera RGB-D modelo IntelRealsense D435 e uma câmera de rastreamento modelo IntelRealsense T265. Todos os sensores tem pacotes distribuídos e bem documentados, tornando fácil a utilização com ROS.

1.3 Objetivo

Nesta dissertação apresenta-se uma metodologia para transposição de escadas completa, abordando (i) a identificação da escada no ambiente, (ii) o gerenciamento dos braços durante as transições entre escada e andares, garantindo uma movimentação suave e livre de colisões além de (iii) seguimento de caminho.

Para identificação das escadas, considera-se que o robô está equipado com uma câmera RGB-D. Elas tem se tornado cada vez mais comuns, baratas e tornaram-se sensores primários na maioria das aplicações robóticas [3]. São por via de regra uma solução sem partes móveis, de tamanho discreto e de baixo consumo energético comparado a lasers rotativos. Elas combinam informações visuais, geralmente vindas de uma câmera RGB com informações de um outro tipo de sensor capaz de medir distâncias (como sensores de distância à laser, scanners de luz estruturada e câmeras infra-vermelho). As features são detectadas por meio de uma sequência de filtros aplicados tanto ao canal de cor quanto de distância: um pré-processamento, feito utilizando um Canny Filter [7] para extração de bordas, um processamento por RANSAC [15] para detecção das linhas, e pós-processamento por RANSAC para detecção do plano que intercepta a borda dos degraus. Após a extração das features, tem-se as linhas ótimas das bordas dos degraus e um plano ótimo passando por todas as bordas. A escada pode então ser modelada em termos de seu início, largura, inclinação e fim.

A estratégia de posições predefinidas para os braços, presente em [28], [27], [32] e outros, não pode ser aplicada ao caso pelos motivos previamente apresentados. O ques-

[1] Página oficial: Robot Operating System

tionamento segue: como mover os braços nas transições de entrada e saída da escada? - A solução adotada neste trabalho é entender o robô como um manipulador e a escada como um plano. A proposta admite o conhecimento da orientação do robô, que pode ser fornecida com certa precisão por uma IMU, o conhecimento das posições de junta e o pitch da escada. Uma análise cinemática é feita afim de obter-se um modelo de equações diferenciais para o sistema. Com as equações, pode-se então entender a subida de escadas como um problema de controle, bastando então projetar os controladores para os braços.

O rumo do robô também é uma preocupação em toda a literatura apresentada. Vários dos autores propõe controladores com dois diferentes objetivos: manter o robô orientado na direção dos degraus e/ou afastado dos limites laterais da escada. Em [36] os autores argumenta que resultados experimentais mostram que um simples controle proporcional é capaz de manter o robô na direção dos degraus até o fim da escada, mas formalmente não há garantias quanto manter-se afastado das bordas. Em [29] os autores projeta um controle por realimentação de estados e usa como referência para o rumo uma razão normalizada entre a distância do robô até os extremidades da escada à esquerda e à direita, mantendo-o de fato afastado das bordas, mas sem garantir um caminho pelo centro da escada. Neste trabalho propõe-se uma lei de controle para seguimento de caminho, onde o caminho é parametrizado passando pelo ponto médio de todos degraus.

1.4 Organização do texto

A organização do texto a seguir se dá em capítulos sobre cada um dos pontos da metodologia proposta. Em primeiro momento o método para detecção e modelagem da escada é revisto, então os conceitos e algoritmos são apresentados junto aos resultados que a validam ao fim. Em sequência, a metodologia de reconfiguração contínua das pernas robóticas para executar a locomoção sobre a escada durante a subida e descida é apresentada para cada etapa definida: transição do solo para escada, locomoção sobre a escada e aterrissagem suave no piso superior. Ao final apresentam-se as conclusões gerais sobre o estudo desenvolvido e uma seção com proposta de trabalhos futuros.

Capítulo 2

Metodologia para modelagem e detecção de escadas

A modelagem e detecção de escadas é um problema cuja solução não é única e depende, em primeiro do momento, do tipo de sensor empregado e fatores ambientais, como iluminação, além do tipo da escada. Na literatura, as escadas são classificadas por duas características: sua inclinação e o traçado definido para circulação de pessoas [8]. Os elementos que constituem a escada possuem uma nomenclatura oriunda da construção civil. As partes frontal e superior dos degraus são chamadas de espelho e piso, respectivamente. Nesta dissertação, a detecção de escadas se limita às escadas com inclinação de 18 a 45 graus e com traçado reto entre pavimentos.

As alternativas para sensoreamento também são diversas. Na literatura apresentada, é possível notar uma preferência da comunidade científica por sensores como lasers, câmeras monoculares e câmeras de profundidade. Neste capítulo é proposto um método para detecção e modelagem de escadas utilizando uma câmera de profundidade e a pointcloud co-registrada por ela fornecida. Pointcloud é um formato de dados que consiste num conjunto denso de pontos, cada qual com sua respectiva coordenada cartesiana e, opcionalmente, uma cor, como na figura 2.1. O uso de câmeras 3D e pointclouds na modelagem

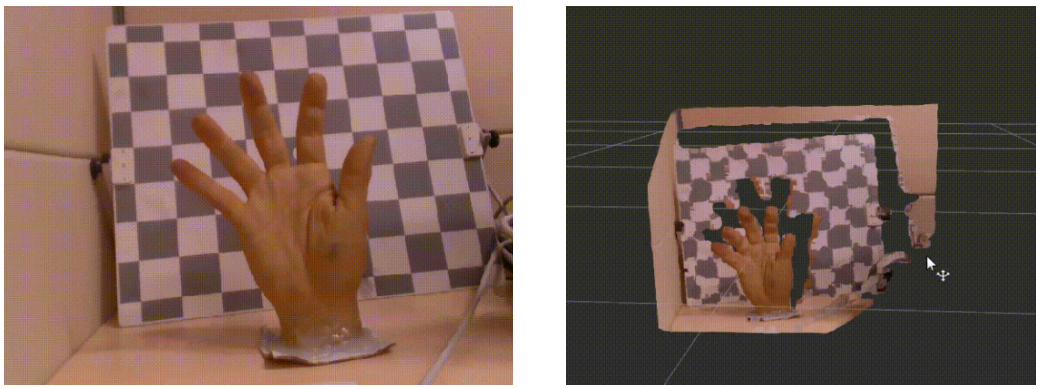


Figura 2.1: À esquerda, a imagem 2D de uma mão. Já à direita, uma representação da mesma cena em pointcloud colorida, fornecida por uma câmera Intel RealSense D435.

de terrenos irregulares é comum na literatura. Recentemente, em [22], o mesmo conjunto proposto neste trabalho é utilizado para modelagem de terrenos irregulares. Os autores faz uso de filtros espaciais, convoluindo a pointcloud com um kernel para erosão e dilatação, removendo assim os pontos esparsados. Em seguida utiliza um operador *Sobel* para modelar os planos da escada e classificar a transponibilidade de uma determinada seção do terreno. Em [23], o autor faz a detecção dos pisos dos degraus utilizando RANSAC, como pode ser visto na figura 2.2. Em [23] e [22], os quadrúpedes aplicados precisam de um modelo rico dos degraus, uma vez que os pontos de apoio das pernas devem ser planejados previamente para o cálculo da trajetória.

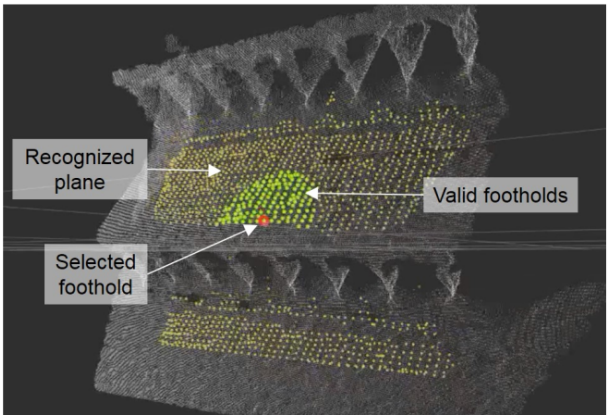


Figura 2.2: Planos detectados e o ponto de apoio para as pernas em [23].

Entretanto, no caso dos AATVs, o modelo da escada não necessita propriamente de uma rica modelagem dos degraus. Ao deslocar-se sobre a escada, o robô em questão movimenta-se sobre um plano inclinado formado pelas bordas dos degraus, e portanto, modelagem deve ser feita de uma forma diferente. As características a serem identificadas são: o início e fim da escada, a largura dos degraus e o plano de movimentação do robô.

Nesta dissertação, o objetivo é utilizar uma sequência de filtros de forma a explorar todos os canais de informação disponíveis: a cor de cada ponto e sua respectiva coordenada cartesiana. Como em [11], o método proposto consiste num pré-processamento por um filtro Canny, com o objetivo de extrair as bordas da imagem, reduzindo enormemente o número de pontos para processamento. Em seguida, uma etapa de processamento, onde aplica-se RANSAC para detecção das retas de interesse nas bordas detectadas. Posteriormente, os pontos das linhas pertencentes à escada são agrupados em um *cluster* e o plano de movimentação é calculado através de uma nova etapa de filtragem por RANSAC e ajuste por mínimos quadrados. Esta última torna possível a determinação de um sistema de coordenadas ao início e fim da escada. Cada uma das etapas será detalhadamente apresentada nas seções seguintes.

2.1 Detecção de bordas

Proposto em [7], o Filtro Canny é uma poderosa técnica de processamento de imagens para extração de bordas. As etapas principais são suavização da imagem através da convolução com um kernel Gaussiano, e então, a análise do gradiente em função de sua norma e direção para detecção das bordas. As aproximações do gradiente são normalmente calculadas através dos operadores *Sobel* ou *Prewitt*, adequados à escalas monocromáticas de cor. Note portanto, que há uma necessidade de adequação, visto que a saída da câmera 3D possui canais RGB e a entrada do filtro deve ter um único canal monocromático. Desta forma, em um primeiro momento é necessário efetuar a conversão de RGB para a escala de cinza. Existem diversas estratégias para esta conversão que vão desde uma simples média da soma dos três canais, à uma média ponderada, com maior peso para os canais cujos comprimentos de onda melhor estimulam o olho humano. Este tópico não será tratado neste trabalho, tendo em vista a vasta gama de bibliotecas que podem ser empregadas para facilitar o processo.

Operações espaciais lineares

Para a implementação do filtro, é necessário que antes os operadores sejam definidos. Na figura 2.3, sejam $f(x,y)$ e $w(s,t)$ respectivamente uma imagem de entrada e um kernel genéricos. Um filtro espacial linear escreve a imagem de saída $g(x,y)$ como uma soma de produtos entre f e w. Em [18], o filtro espacial linear de um imagem de tamanho $M \times N$

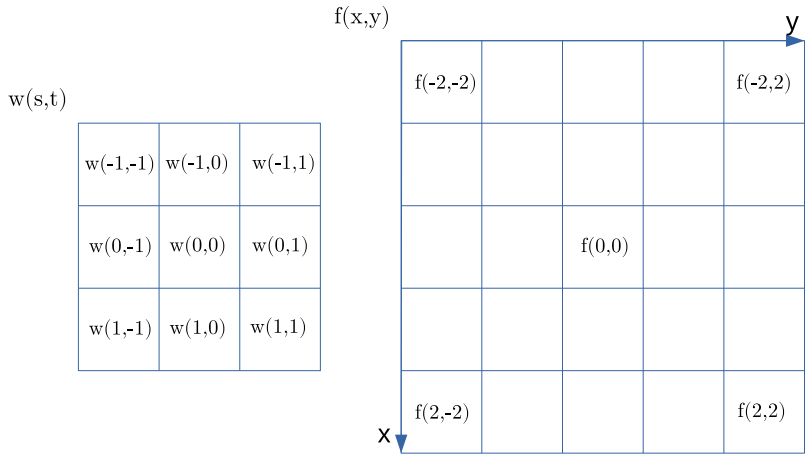


Figura 2.3: O kernel w à esquerda e a imagem f à direita. Note que os índices são contados de cima para baixo, da esquerda para a direita.

com um kernel de tamanho $m \times n$ é definido como

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t), \tag{2.1}$$

onde, de forma geral, $m = 2a + 1$ e $n = 2b + 1$, sendo a e b inteiros não negativos, garantindo portanto um tamanho ímpar tanto em colunas quanto em linhas.

Correlação e convolução espaciais

Na figura 2.4, a correlação consiste em mover o kernel ao longo de uma imagem, computando a soma de produtos para cada um dos *pixels*. De forma análoga, a convolução consiste em mover o kernel a longo de uma imagem rotacionada por um valor de 180 graus. Portanto, ambas as operações apresentam o mesmo resultado quando kernels simétricos em relação ao centro são usados.

A equação (2.1) é a correlação espacial de um kernel w com uma imagem f e é denotada por

$$w(s,t) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t). \tag{2.2}$$

Por outro lado, a convolução espacial de um kernel w com uma imagem f é denotada por

$$w(s,t) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t). \tag{2.3}$$

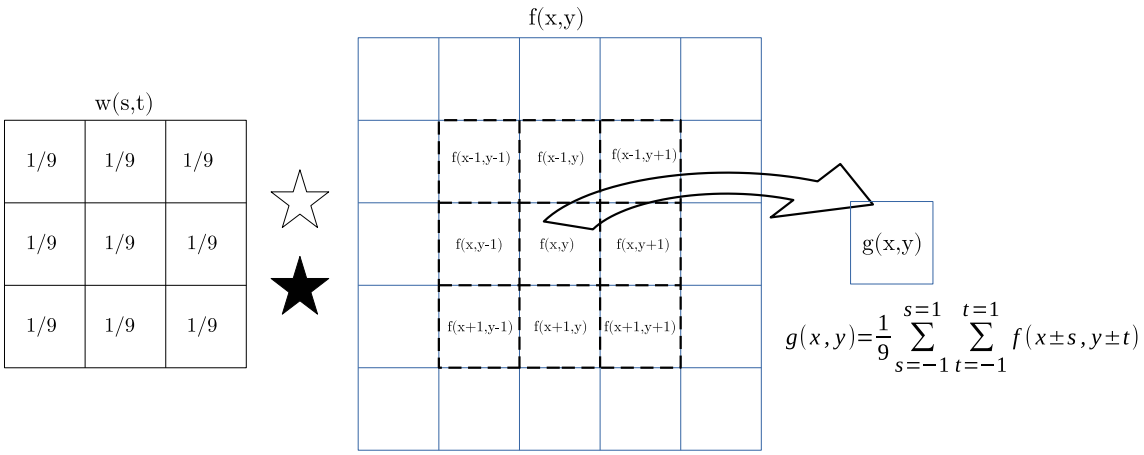


Figura 2.4: A figura mostra uma operação entre o kernel $w(s,t)$ e o *pixel* $f(x,y)$ da imagem. Tanto a correlação quanto a convolução levam a um mesmo resultado, visto que w é simétrico com relação ao centro. A filtragem espacial linear com este kernel é conhecida como filtro caixa. Basicamente este filtro suaviza a imagem, atribuindo a cada *pixel* uma média entre o mesmo e todos o *pixels* vizinhos.

Filtros gaussianos

Os filtros caixa, como os da figura 2.4, são importantes por sua simplicidade, oferecendo muitas vezes resultados visualmente aceitáveis [18]. Entretanto, quando aplicado à imagens com alto nível de detalhes ou componentes geométricos fortes, a direcionalidade do

filtro produz resultados pouco satisfatórios. Para este tipo de aplicação usam-se filtros gaussianos por seu melhor desempenho. Um kernel para este tipo de filtro é dado por uma função gaussiana do tipo

$$w(s,t) = G(s,t) = k\,e^{-\frac{s^2+t^2}{2\sigma^2}} \tag{2.4}$$

onde k é uma constante e σ é a dispersão, variando a forma do sino. Note que no caso de uma função de densidade de probabilidade, $k = \frac{1}{2\sigma^2\,\pi}$ e σ é a variância. Contudo, para o emprego do filtro, apenas a forma de sino da função gaussiana é importante, não estando k necessariamente restrito em função de σ.

A forma quadrática definida negativa no expoente do número de Euler faz com que o kernel tenha seu maior valor em $(0,0)$. A medida que s e t se afastam da origem, o valor de $w(s,t)$ diminui, havendo portanto, pouco sentido utilizar um kernel de grandes dimensões visto que os termos afastados da origem tem efeito quase nulo. A depender de σ, em geral, usam-se *kernels* de tamanho $[6\sigma] \times [6\sigma]$. Como procura-se trabalhar com dimensões ímpares, usa-se o menor número inteiro possível de forma que a condição seja alcançada. Isto é: para $\sigma = 7$, deve-se usar um kernel de dimensões 43×43.

$\mathrm{w(s,t)}$

$$\frac{1}{4.8976} \times \begin{array}{|c|c|c|} \hline 0.3679 & 0.6065 & 0.3679 \\ \hline 0.6065 & 1.0 & 0.6065 \\ \hline 0.3679 & 0.6065 & 0.3679 \\ \hline \end{array}$$

Figura 2.5: A figura mostra um kernel gaussiano com $k = \sigma = 1$. Note que a dimensão apropriada para $w(s,t)$ é de 7×7. Dimensões reduzidas não são capazes de reproduzir o formato de sino da curva gaussiana, comprometendo os resultados.

O gradiente de uma imagem

O gradiente de uma imagem $f(x,y)$ é definido como o vetor coluna

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{df}{dx} \\ \frac{df}{dy} \end{bmatrix}. \tag{2.5}$$

Portanto, para obter o gradiente da imagem é necessário o cálculo das derivadas parciais de f. Uma definição para as derivadas de primeira ordem da função em questão pode ser obtida por um operador de diferenças de primeira ordem

$$\frac{df}{dx} = f(x+1, y) - f(x, y) \tag{2.6}$$

e

$$\frac{df}{dy} = f(x, y+1) - f(x, y). \tag{2.7}$$

As diferenças podem ser calculadas para cada um dos *pixels* através do kernel da figura 2.6. Entretanto, a aplicação deste kernel para obtenção das derivadas não se comporta muito bem com imagens ruidosas, além de desconsiderar os *pixels* em direções diagonais. O problema está na aproximação discreta das derivadas em (2.6) e (2.7) usando um operador de diferenças finitas de primeira ordem.

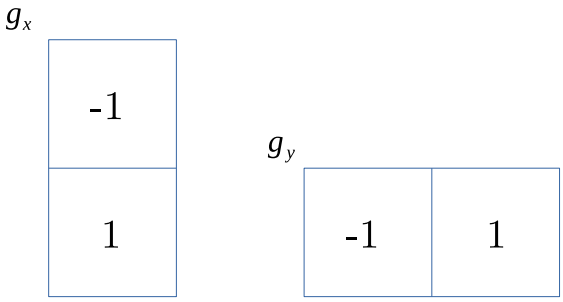


Figura 2.6: A figura mostra o kernel mais simples para obtenção das derivadas parciais de uma função, servindo apenas para fins elucidativos.

Por vezes, no cálculo de aproximações discretas de derivadas, o uso de operadores de diferenças finitas de segunda ordem é mais vantajoso. O erro deste tipo de operadores decresce por uma razão de Δx^2, sendo portanto, mais adequado. Na figura 2.7, o kernel Sobel é o mais utilizado para cálculo do gradiente. Ele faz o uso de um operador de diferenças finitas de segunda ordem, calculando as derivadas utilizando os *pixels* anterior e posterior ao analisado. Além disso, ele considera os *pixels* nas diagonais, atribuindo um peso ligeiramente maior nas direções principais

$$g_x = \frac{df}{dx} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \tag{2.8}$$

e

$$g_y = \frac{df}{dy} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7). \tag{2.9}$$

f(x,y)

z1	z2	z3
z4	z5	z6
z7	z8	z9

g_x

-1	-2	-1
0	0	0
1	2	1

g_y

-1	0	1
-2	0	2
-1	0	1

Figura 2.7: Em conjunto com as equações (2.8) e (2.9), a figura mostra o cálculo das derivadas parciais com operadores Sobel. Note o valor nulo na direção dos eixos principais, característicos da diferença finita de segunda ordem.

Filtro Canny

Seja $f(x,y)$ uma imagem de entrada. A solução proposta em [7] para extração das bordas em f pode ser resumida em quatro procedimentos. O primeiro deles é a suavização da imagem, geralmente realizada com filtro gaussiano apresentado em (2.4). Deste modo, forma-se a imagem $f_s(x,y)$, definida como a convolução de f com um kernel gaussiano G em

$$f_s(x,y) = G(w,s) \star f(x,y). \tag{2.10}$$

A segunda etapa consiste aplicar em $f_s(x,y)$ os operadores (geralmente Sobel) para cálculo da norma $M_s(x,y)$ do gradiente e seu ângulo $\alpha_s(x,y)$ com relação ao eixo x, definidos como

$$M_s(x,y) = \sqrt{{g_x}^2(x,y) + {g_y}^2(x,y)} \tag{2.11}$$

e

$$\alpha_s(x,y) = atan2\left(\frac{g_y(x,y)}{g_x(x,y)}\right), \tag{2.12}$$

onde $g_x(x,y)$ e $g_y(x,y)$ podem ser calculados como em (2.8) e (2.9). A imagem formada pela norma do gradiente $M_s(x,y)$ normalmente contém bordas largas, e portanto, a próxima etapa tem o objetivo de afiná-las.

Na terceira etapa utiliza-se então, um método chamado de *nonmaxima suppression*. Considere um *pixel* (x,y). Este método consiste em analisar a intensidade dos *pixels* vizinhos na direção do gradiente, e com base na análise, decidir se (x,y) deve ser suprimido (apagado) ou não. Na figura 2.8, considere as discretizações de direção d_1, d_2, d_3 e d_4 para o ângulo α_s. Seja $g_n(x,y)$ a imagem de saída. Para um *pixel* (x,y) qualquer, o método pode ser formulado da seguinte forma:

1. Encontrar a direção d_k correspondente ao $\alpha_s(x,y)$;

2. Defina $k = M_s(x,y)$. Se k é menor que a norma do gradiente M_s para um ou ambos os vizinhos na direção d_k do *pixel* (x,y), $g_n(x,y) = 0$. Se não, $g_n(x,y) = k$.

A quarta e última etapa tem como finalidade reduzir o número de bordas falsas. A

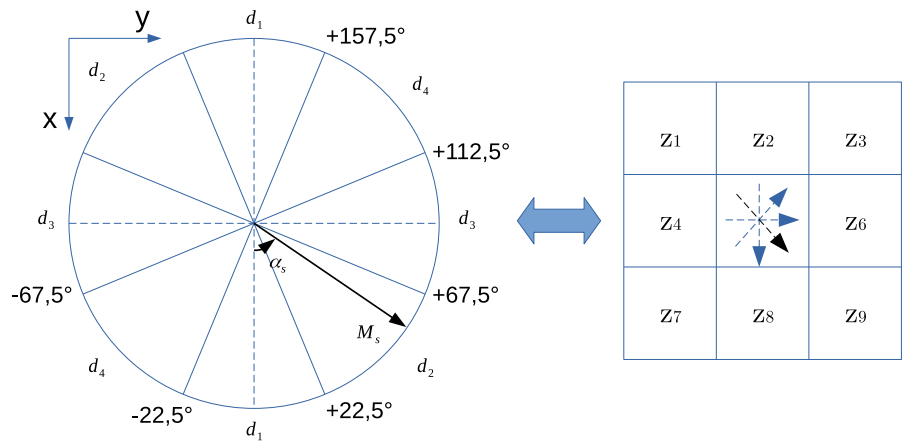


Figura 2.8: Na figura, M_s representa a norma do gradiente, α_s seu ângulo com relação ao eixo x e d_1, d_2, d_3 e d_4 são as direções de busca em função de α_s. Note que no caso, α está numa faixa de valores entre 22,5° e 67,5°. Portanto, a direção de busca é d_2 e os vizinhos sob análise são z_1 e z_9.

etapa é chamada de *thresholding* e usa dois valores de *threshold* sendo um superior, T_h, e um inferior, T_l. Considere então as imagens

$$g_{nh}(x,y) = g_n(x,y) \geq T_h$$

e

$$g_{nl}(x,y) = g_n(x,y) \geq T_l.$$

Inicialmente, g_{nh} e g_{nl} são inciadas com todos *pixels* iguais a zero. Após o *thresholding*, num primeiro momento, todos os *pixels* em g_{nh} estarão também em g_{nl}. Desta forma, apagam-se de g_{nl} todos os *pixels* contidos em g_{nh} com

$$g_{nl}(x,y) = g_{nl}(x,y) - g_{nh}(x,y).$$

Todos os *pixels* não-nulos de g_{nh} são considerados bordas. O procedimento segue:

1. Para cada *pixel* não-nulo em g_{nh}, busca-se na vizinhança do mesmo *pixel* em g_{nl} por *pixels* não-nulos que estejam a ele conectados;

2. Todos os *pixels* encontrados anteriormente são adicionados à g_{nh}.

Ao final, obtém-se uma imagem $g_{nh}(x,y)$ com os melhores candidatos às bordas da imagem de entrada $f(x,y)$.

2.2 Extração de retas das bordas

Ao final da filtragem descrita na etapa anterior, a pointcloud filtrada contém apenas pontos que pertencem às melhores candidatas à bordas, corretamente ou incorretamente detecta-

das. Cabe ressaltar ainda, que nem todas as bordas detectadas são pertencentes à escada, podendo ser portas, portais ou qualquer outro objeto cuja geometria tenha uma variação brusca do gradiente. Uma nova etapa de filtragem conhecida na literatura como RANSAC [15] (acrônimo para *Random Sample Consensus*) é aplicada aos pontos das bordas candidatas. Desta vez entretanto, usando sua posição cartesiana com relação à um sistema de coordenadas qualquer. Esta etapa tem o objetivo de separar os melhores pontos pertencentes às bordas, para posteriormente utilizá-los na parametrização de retas. Com as retas parametrizadas, realiza-se então a seleção do conjunto de retas de interesse, isto é: retas cujos parâmetros sejam coerentes com os de uma escada.

A figura 2.9 apresenta um conjunto de pontos uma pointcloud $\chi = \{p_1, p_2, \cdots, p_n\}$, onde n é o número total de pontos e $p_i \in \mathbb{R}^3$, $i = 1, 2, \cdots, n$. As bordas detectadas anteriormente são parametrizadas por retas, e portanto, seja $\alpha = 2$ o menor número de pontos necessários para traçar uma reta. Sejam $\{p_{k_1}, p_{k_2}\} \in \chi$ os α pontos aleatoriamente sorteados numa dada iteração k. Considere uma reta entre os pontos sorteados. A distância do ponto p_i à esta reta pode ser calculada da seguinte forma: seja

$$\rho_k = p_{k_2} - p_{k_1} \tag{2.13}$$

e

$$e_i = p_i - p_{k_1}. \tag{2.14}$$

Sabe-se que

$$e_{\perp_i} = e_i - \rho_k \sigma_i, \tag{2.15}$$

onde $\sigma_i \in \mathbb{R}$ é uma distância na direção de ρ_k. Entretanto, note que

$$\rho_k{}^T e_{\perp_i} = 0 \tag{2.16}$$

e portanto, tem-se que

$$\sigma_i = \frac{\rho_k{}^T e_i}{\rho_k{}^T \rho_k}. \tag{2.17}$$

Desta forma, substituindo (2.17) em (2.15), chega-se em

$$e_{\perp_i} = \left(I_{3\times 3} - \frac{\rho_k \rho_k{}^T}{\rho_k{}^T \rho_k} \right) e_i \tag{2.18}$$

e a distância de um ponto p_i até a reta traçada entre os pontos $\left(p_{k_1}, p_{k_2}\right)$ é

$$d_i = \left\| e_{\perp_i} \right\|. \tag{2.19}$$

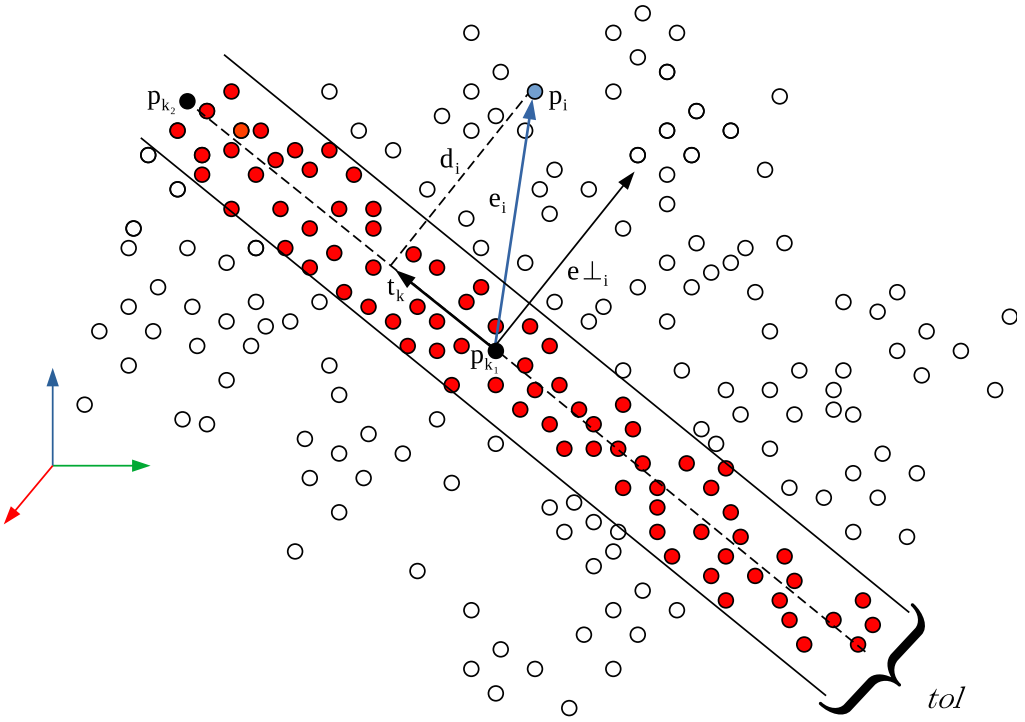


Figura 2.9: A figura mostra uma determinada iteração k do RANSAC. O i-ésimo ponto é classificado em função de sua distância d_i até a reta formada entre os pontos $\left(p_{k_1}, p_{k_2}\right)$. Os pontos em vermelho ($d_i < tol$) são *Inliers*, enquanto os outros ($d_i \geq tol$), sem preenchimento, são os *Outliers*.

Para uma dada iteração k, definam-se $Inliers_k$ e $Outliers_k$ dois subconjuntos de χ

$$Inliers_k = \left\{p_i \,|\, d_i\left(p_i\right) < tol, \quad i = 1, 2, \cdots, n\right\} \tag{2.20}$$

e

$$Outliers_k = \left\{p_i \,|\, d_i\left(p_i\right) \geq tol, \quad i = 1, 2, \cdots, n\right\}, \tag{2.21}$$

onde $tol \in \mathbb{R}$ é uma tolerância arbitrariamente definida.

O método proposto em [15] consiste em iterar k vezes na busca dos pontos $\left\{p_{k_1}, p_{k_2}\right\}$ que contém o maior número de elementos no subconjunto dos $Inliers$. O número máximo de iterações k_{max} é definido para satisfazer critérios de desempenho probabilísticos: k_{max} deve ser tal que garanta uma probabilidade de que o melhor subconjunto de $Inliers$ será sorteado.

Seja ϵ uma estimativa prévia da proporção de $Outliers$ na população total. Portanto, a probabilidade de um ponto p pertencer ao conjunto dos $Outliers$ numa dada iteração é ϵ. De certo, $1 - \epsilon$ é a probabilidade de p pertencer ao conjunto dos $Inliers$. Logo, para α pontos aleatoriamente amostrados, a probabilidade de $\left\{p_1, p_2, \cdots, p_\alpha\right\}$ pertencerem ao conjunto dos $Inliers$ é de $(1 - \epsilon)^\alpha$. Complementarmente, a probabilidade de pelo menos um dos pontos pertencerem ao conjunto dos $Outliers$ é de $1 - (1 - \epsilon)^\alpha$. Se o processo

for repetido N vezes, tem-se que $\left(1-(1-\epsilon)^{\alpha}\right)^{N}$ é a probabilidade de, senão todos, pelo menos um ponto, pertencer ao subconjunto dos $Outliers$. Seja P a probabilidade de que todos os pontos pertençam ao subconjunto dos $Inliers$. Pode-se escrever então que $\left(1-(1-\epsilon)^{\alpha}\right)^{N} = (1-P)$. Logo,

$$N = \frac{\log(1-P)}{\log\left(1-(1-\epsilon)^{\alpha}\right)}. \tag{2.22}$$

Portanto, usando a equação (2.22) pode-se determinar $k_{max} = N$, o número de iterações necessárias para uma probabilidade P de que o melhor conjunto de $Inliers$ seja encontrado. Na implementação, de forma geral usa-se P alto ($P \geq 0.95$). Normalmente, como ϵ é desconhecido a priori, assume-se inicialmente um valor próximo de 1 e a razão entre número de pontos $Outliers$ e a população total é atualizada a cada iteração. Deste modo, o número de iterações necessárias calculado na equação (2.22) é atualizado iterativamente. Ao final, o conjunto de $Inliers$ com maior número de elementos será provavelmente a melhor reta no conjunto de pontos χ.

Note entretanto que o método descrito aplica-se a casos unimodais, onde somente há uma *feature* a ser detectada. Em outras palavras: há somente uma borda no conjunto de pontos χ da pointcloud. Para aplicações multimodais, aplica-se o algoritmo descrito de forma sequencial. Ao final da extração de cada *feature*, seus respectivos melhores pontos $Inliers$ são removidos do espaço amostral e o algoritmo é repetido até que não haja mais um conjunto de pontos candidato à uma reta. Por fim, chega-se à um conjunto $\Lambda = \{\lambda_1, \lambda_2, \cdots, \lambda_m\}$, onde m é o número de retas detectadas. Cada elemento λ do conjunto Λ contém os melhores pontos $Inliers$ de uma reta correspondente. A parametrização de cada uma das retas em Λ é feita por meio do ajuste dos pontos por Mínimos Quadrados.

Ajuste de retas 3D por Mínimos Quadrados

No espaço 3D, retas não podem ser descritas por uma única equação linear, visto que estas formam planos. Na figura 2.10, uma forma clássica de parametrizá-las é pelo conjunto de equações

$$\begin{cases} x = x_0 + k_x\, t \\ y = y_0 + k_y\, t \\ z = z_0 + k_z\, t \end{cases}, \tag{2.23}$$

onde $t \in \mathbb{R}$ é uma variável livre, $v = \begin{bmatrix} k_x & k_y & k_z \end{bmatrix}^T$ é o vetor diretor e $\left(x_0, y_0, z_0\right)$ é um ponto arbitrário sobre a reta.

Como na figura 2.11, retas 3D também podem ser descritas como a interseção de dois

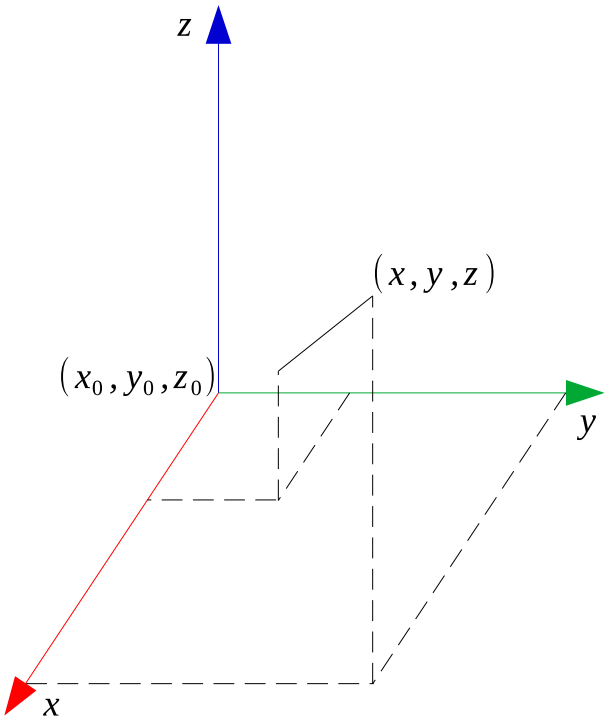


Figura 2.10: Breve seção da reta parametrizada pela equação (2.23) entre o ponto (x_0, y_0, z_0) e (x, y, z).

planos. Ao manipular a equação (2.23) tem-se que

$$\frac{x - x_0}{z - z_0} = \frac{k_x}{k_z}$$

e

$$\frac{y - y_0}{z - z_0} = \frac{k_y}{k_z}.$$

Portanto, pode-se escrever

$$\begin{cases} x = a_r\, z + b_r \\ y = c_r\, z + d_r \end{cases}, \tag{2.24}$$

onde os coeficientes $a_r = \frac{k_x}{k_z}$, $b_r = x_0 - \frac{k_x}{k_z}\, z_0$, $c_r = \frac{k_y}{k_z}$ e $d_r = y_0 - \frac{k_y}{k_z}\, z_0$.

Considere então os pontos contidos em um dado elemento do conjunto Λ e o sistema de equações (2.24). Pode-se calcular os parâmetros (a_r, b_r, c_r, d_r) utilizando a técnica dos Mínimos Quadrados. Seja m o número de pontos em uma reta $\lambda \in \Lambda$ e (x_i, y_i, z_i) $i = 1, 2, \cdots, m$ suas coordenadas cartesianas, é possível escrever

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} z_1 & 1 \\ z_2 & 1 \\ \vdots & \vdots \\ z_n & 1 \end{bmatrix} \begin{bmatrix} a_r \\ b_r \end{bmatrix} \tag{2.25}$$

e

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} z_1 & 1 \\ z_2 & 1 \\ \vdots & \vdots \\ z_n & 1 \end{bmatrix} \begin{bmatrix} c_r \\ d_r \end{bmatrix}. \tag{2.26}$$

As equações (2.25) e (2.26) podem ser reescritas como

$$X = A_r \beta_x \tag{2.27}$$

e

$$Y = A_r \beta_y. \tag{2.28}$$

Logo, tem-se que

$$\beta_x^* = \left(A_r^T A_r\right)^{-1} A_r^T X \tag{2.29}$$

e

$$\beta_y^* = \left(A_r^T A_r\right)^{-1} A_r^T Y. \tag{2.30}$$

onde o sobrescrito $*$ sinaliza que $\beta_x^* = [a_r^* \quad b_r^*]^T$ e $\beta_x^* = [c_r^* \quad d_r^*]^T$ são parâmetros otimizados, visto que A_r não tem posto completo. Note ainda, que a reta da equação

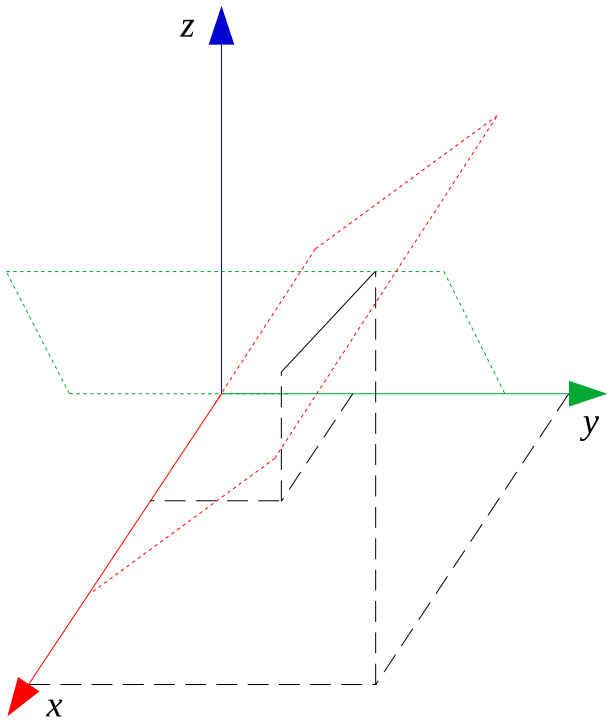


Figura 2.11: Na figura, a reta é entendida como a interseção de dois planos, um em verde e outro em vermelho. O plano em verde é parametrizado pela equação $x = a_r z + b_r$ e corre ao longo do eixo y de $-\infty$ à ∞. Já o plano em vermelho é parametrizado pela equação $y = c_r z + d_r$ e corre ao longo do eixo x de $-\infty$ à ∞.

(2.23) é a interseção dos dois planos $-x + a_r^* z + b_r^* = 0$ e $-y + c_r^* z + d_r^* = 0$ cujos vetores normais são conhecidos:

$$\eta_x = \begin{bmatrix} -1 \\ 0 \\ a_r^* \end{bmatrix} \tag{2.31}$$

para a primeira equação e

$$\eta_y = \begin{bmatrix} 0 \\ -1 \\ c_r^* \end{bmatrix} \tag{2.32}$$

para a segunda. Portanto, o vetor diretor da reta é

$$\nu_{\parallel} = \begin{bmatrix} k_x \\ k_y \\ k_z \end{bmatrix} = \hat{\eta}_x \, \eta_y \tag{2.33}$$

onde o operador $\hat{\eta}$ denota produto vetorial. Isto é

$$\hat{\eta} = \eta \times = \begin{bmatrix} 0 & -\eta_3 & \eta_2 \\ \eta_3 & 0 & -\eta_1 \\ -\eta_2 & \eta_1 & 0 \end{bmatrix}. \tag{2.34}$$

Ao repetir o processo para cada um dos elementos de Λ, todas as retas são parametrizadas. Com os parâmetros é possível selecionar quais das retas tem padrões semelhantes ao de uma escada. O comprimento das retas pode ser encontrado buscando o ponto mais afastado nos dois sentidos do vetor diretor $\nu_{\parallel}$.

Filtragem das retas de interesse

Defina-se o vetor $v_{\perp}$ como a componente ortogonal de um vetor v em relação à um determinado eixo e, i.e.

$$v_{\perp} = \left(I_{3\times 3} - ee^T \right) v.$$

O conjunto de retas de interesse Y é um subconjunto de Λ com retas que obedecem aos seguintes critérios:

1. o número de pontos contidos na reta deve ser maior que uma quantidade mínima c_m (removendo linhas com baixo número de elementos);

2. o comprimento minimo, definido como a norma entre a posição dos dois pontos limítrofes, deve ser maior que um comprimento l_m (removendo linhas cujo comprimento não é condizente com a borda de um degrau);

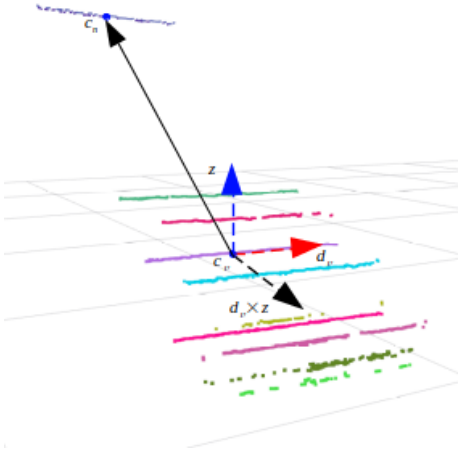


Figura 2.12: Na figura, note as múltiplas retas detectadas no mesmo piso e/ou espelho.

3. seja d um vetor com a direção de uma reta e $d_\perp$ a componente perpendicular ao vetor normal ao plano do andar inferior. O ângulo α entre d e $d_\perp$ deve satisfazer a condição $|\alpha| < tol_\alpha$ (removendo linhas com elevação incoerente);

4. seja β o ângulo entre $d_\perp$ e o vetor normal a face do sensor RGB-D. A condição $|\beta| > tol_\beta$ deve ser satisfeita (removendo linhas não horizontais);

As retas candidatas são agrupadas em um novo conjunto $\mathrm{Y} = \{v_1, v_2, \cdots, v_l\}$ onde l é o número de retas selecionadas.

Como apresentado na figura 2.12, ainda que Y contenha todas as restas das bordas dos degraus, múltiplas retas detectadas em um mesmo espelho ainda necessitam ser filtradas. Desta forma, a seguinte sequência de operações é executada.

1. Seja $c_{v_i} \in \mathbb{R}^3$ a posição do centroide da iésima reta em Y;

2. Seja $d_{v_i} \in \mathbb{R}^3$ a direção da reta v_i;

3. Define-se $n_{v_i} = d_{v_i} \times z$ o vetor normal ao plano entre d_{v_i} e z;

4. Seja $e_{ij} = c_{v_i} - c_{v_j}$, com $j = 1, 2, \ldots, l$ e $i \neq j$ a posição relativa entre centroides;

5. Em uma dada iteração, de todas as retas que respeitam a condição $|n_{v_i}^T e_{ij}| < \mathrm{tol}_{plane}$ (retas coplanares em um mesmo espelho), seleciona-se a que possui o maior centroide na coordenada z. Descartam-se as outras retas e seus respectivos pontos;

2.3 Extração de planos das retas selecionadas

A etapa final da detecção de escadas consiste em modelar o plano que intercepta as bordas de todos os degraus. Embora Y seja composto majoritariamente por retas das bordas

do degraus, é possível que ainda contenha retas incorretamente detectadas. Usa-se novamente RANSAC para detecção de planos no conjunto de pontos das retas contidas em Y. Desta vez entretanto, num enfoque unimodal, onde há somente um plano desejado.

Planos podem ser formados por um conjunto de $\alpha = 3$ pontos. De forma análoga à seção anterior, sejam $\left\{p_{k_1}, p_{k_2}, p_{k_3}\right\} \in Y$ os α pontos aleatoriamente sorteados numa dada iteração k. Considere dois vetores entre os pontos sorteados

$$\rho_{k_1} = p_{k_2} - p_{k_1} \tag{2.35}$$

e

$$\rho_{k_2} = p_{k_3} - p_{k_1}. \tag{2.36}$$

Usando a definição (2.34), pode-se escrever o vetor normal ao plano formado por ρ_{k_2} e ρ_{k_1} como

$$\eta_{\perp} = \hat{\rho}_{k_2} \rho_{k_1}. \tag{2.37}$$

Seja n o número total de pontos no conjunto Y, p_i para $i = 0, 1, \cdots, n$ o i-ésimo ponto do conjunto e d_i a distância entre o ponto p_i e o plano formado por ρ_{k_2} e ρ_{k_1}

$$d_i = |\eta_{\perp}^T \left(p_i - \rho_{k_1}\right)|. \tag{2.38}$$

De forma análoga à detecção de retas, para uma dada iteração k, definam-se $Inliers_k$ e $Outliers_k$ dois subconjuntos de Y

$$Inliers_k = \left\{p_i \,|\, d_i\left(p_i\right) < tol_{dist}, \quad i = 1, 2, \cdots, n\right\} \tag{2.39}$$

e

$$Outliers_k = \left\{p_i \,|\, d_i\left(p_i\right) \geq tol_{dist}, \quad i = 1, 2, \cdots, n\right\}, \tag{2.40}$$

onde tol_{dist} é uma distância pequena de tolerância. O número de iterações necessárias para que se chegue ao melhor conjunto de $Inliers$ com uma probabilidade elevada é determinado pela mesma equação (2.22) descrita na seção anterior.

Com o melhor conjunto de pontos $Inliers$ encontrado no RANSAC para detecção de planos, aplica-se novamente a técnica de Mínimos Quadrados, desta vez ajustando os pontos à um plano, parametrizado pela equação característica

$$a_p x + b_p y + z + d_p = 0,$$

considerando o coeficiente que multiplica o termo z igual a 1. Pode-se escrever então

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} = - \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_m & y_m & 1 \end{bmatrix} \begin{bmatrix} a_p \\ b_p \\ d_p \end{bmatrix} \tag{2.41}$$

para todos os m pontos contidos no melhor conjunto de $Inliers$. A equação (2.41) pode ser reescrita na forma

$$Z = A_p \beta \tag{2.42}$$

e logo

$$\beta^* = \left(A_p^T A_p \right)^{-1} A_p^T Z. \tag{2.43}$$

Note que são conhecidos um vetor normal ao plano

$$\nu_\perp = \begin{bmatrix} a_p \\ b_p \\ 1 \end{bmatrix}, \tag{2.44}$$

um vetor na direção das bordas dos degraus, na equação (2.33), e o terceiro

$$\nu_a = \hat{v}_\perp \nu_\parallel, \tag{2.45}$$

calculado como o produto vetorial entre os dois anteriores. Considere então a figura 2.13 onde

$$x_{s_i} = \frac{\nu_a}{\|\nu_a\|}, \quad y_{s_i} = \frac{\nu_\parallel}{\|\nu_\parallel\|}, \quad z_{s_i} = \frac{\nu_\perp}{\|\nu_\perp\|}. \tag{2.46}$$

Tem-se

$$E_{s_i} = \begin{bmatrix} x_{s_i} & y_{s_i} & z_{s_i} \end{bmatrix} \tag{2.47}$$

o matriz SO(3) com sistema de coordenadas posicionado ao início da escada. O mesmo processo pode ser repetido para a determinação de

$$E_{s_e} = \begin{bmatrix} x_{s_e} & y_{s_e} & z_{s_e} \end{bmatrix}, \tag{2.48}$$

o sistema de coordenadas ao fim da escada, finalizando portanto a modelagem proposta no início do capítulo. O pseudo-código para extração de planos e alocação dos sistemas de coordenadas discutidos é apresentado em 1.

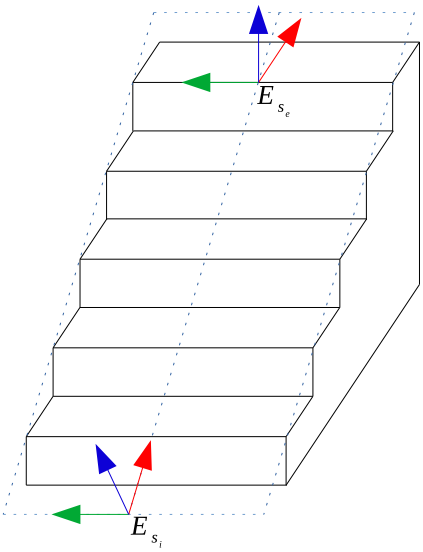


Figura 2.13: A figura mostra uma escada posicionada sobre um plano. No plano anterior à escada, sobre a linha de centro estão posicionados os sistemas de coordenadas inicial E_{s_i} e final, E_{s_e}.

Algorithm 1: Pseudo-código para extração do plano e alocação dos sistemas de coordenadas.

```
Entrada: conjunto de retas filtradas Y;
begin
    Faça RANSAC nas linhas em Y em busca do plano P;
    foreach v ∈ Y do
        Calcule o centróide da linha v;
        Aloque um sistema de coordenadas com origem no centróide, com eixo z
          na direção do vetor normal ao plano P, eixo y na direção da reta e eixo x
          como produto vetorial de y por z;
    end
    Publique os respectivos sistemas de coordenadas;
end
```

2.4 Resultados experimentais

Para validação da metodologia proposta, uma câmera modelo Intel Realsense D435 foi posicionada à frente de escadas com diferentes características para gravação de point-clouds à uma altura coerente com os AATVs abordados neste trabalho (relativamente próxima ao solo). Um pacote ROS foi desenvolvido em C++, fazendo extensivo uso das bibliotecas Point Cloud Library e Eigen para processamento dos dados gravados. O software desenvolvido é utilizado para obtenção da largura dos degraus, do vertor normal ao plano passando pelas bordas dos degraus, além de dois sistemas de coordenadas, um

posicionado ao início e outro ao fim da escada, ambos posicionados ao ponto médio dos degraus como na figura 2.13. As medidas obtidas são então comparadas com os valores nominais obtidos com instrumentação própria.

2.4.1 Escadas detectadas

Em ambas as figuras 2.14 e 2.15 apresentam-se as pointclouds originais e o resultado após as respectivas etapas de filtragem. Após a detecção de bordas, os dados de cor dos pontos se tornam artificiais, de forma que cada uma das linhas detectadas possui uma cor diferente. Em seguida, após a filtragem das linhas com parâmetros de interesse, ajusta-se um plano cujos pontos são coloridos em branco. Cada uma das linhas pertencentes ao plano tem seu centroide calculado e marcado com uma esfera de cor azul. Posteriormente, em cada um dos centroides é alocado um sistema de coordenadas, de modo que os eixos x, y e z estão dispostos como na figura 2.13.

Os resultados foram obtidos através visualizador 3D Rviz junto aos sistemas de coordenadas publicados pelo pacote de ROS desenvolvido. O pitch da escada foi calculado como o ângulo entre o eixo x do sistema de coordenadas *step_1* e a projeção deste mesmo eixo no plano $y = 0$ do sistema de coordenadas *camera_color_optical_frame*. Embora menos importantes para o método proposto neste trabalho, o comprimento da escada e largura dos degraus também puderam ser calculados sem grande esforço. Desta forma, o primeiro foi calculado como a distância entre a origem dos sitemas de coordenadas *step_1* e *step_4* enquanto a largura dos degraus foi calculada como a média aritmética simples do comprimento das linhas pertencentes ao plano dos degraus. Os dados são apresentados nas tabelas 2.1 e 2.2

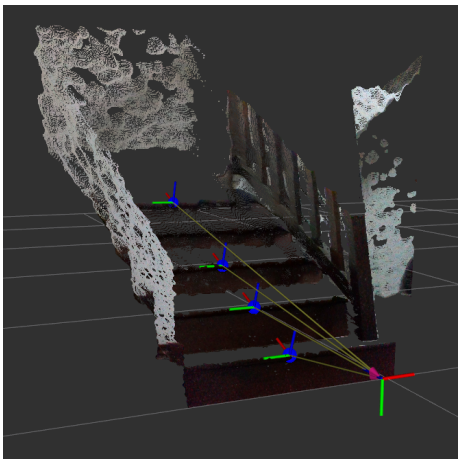
	Valor real	Valor estimado
Pitch com relação ao solo	37.5°	37.3°
Comprimento do plano formado pelas bordas	122cm	122cm
Largura dos degraus	85cm	77cm

Tabela 2.1: Dimensões da escada apresentada na figura 2.14.

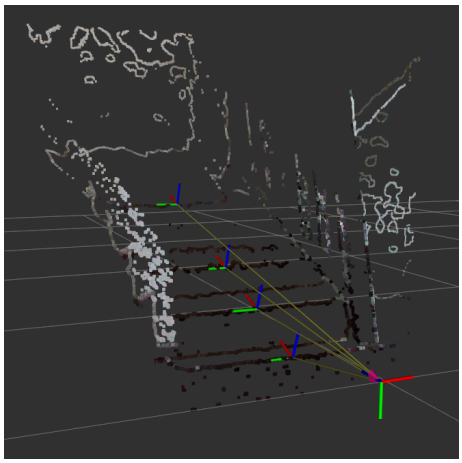
	Valor real	Valor estimado
Pitch com relação ao solo	37.2°	37.2°
Comprimento do plano formado pelas bordas	230cm	226cm
Largura dos degraus	80cm	78cm

Tabela 2.2: Dimensões da escada apresentada na figura 2.15.

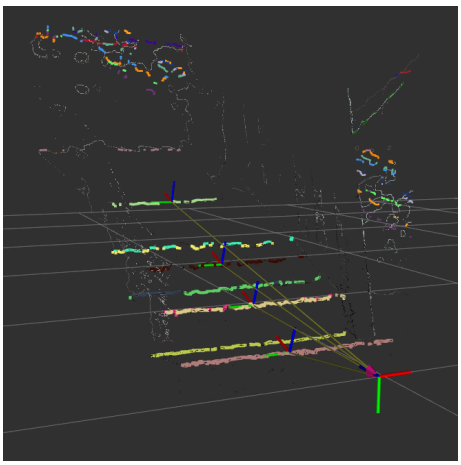
Nota-se que conforme as bordas se afastam do sensor RGB-D, há um aumento tanto no nível de ruído das medidas quanto na quantidade de pontos oclusos, de forma que à uma certa distância do sensor, já não é possível diferenciar bordas e linhas. Tal fato se torna mais evidente na figura 2.15(a), visto a incapacidade do algoritmo de identificar novas bordas dos degraus. É possível associar a largura dos degraus medida nas tabelas



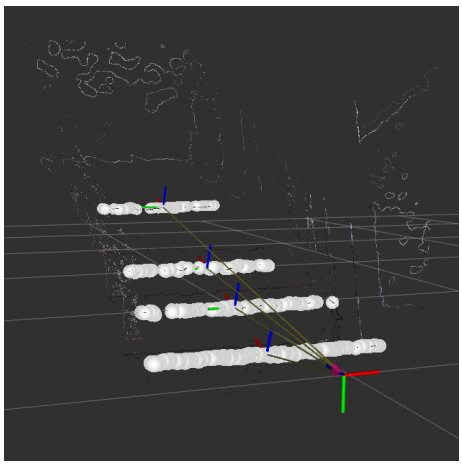
(a) Nuvem de pontos original.



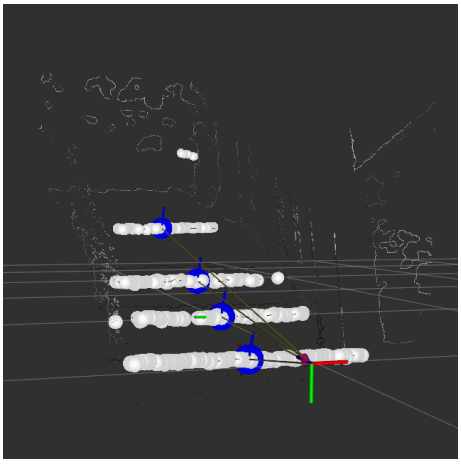
(b) Detecção de bordas.



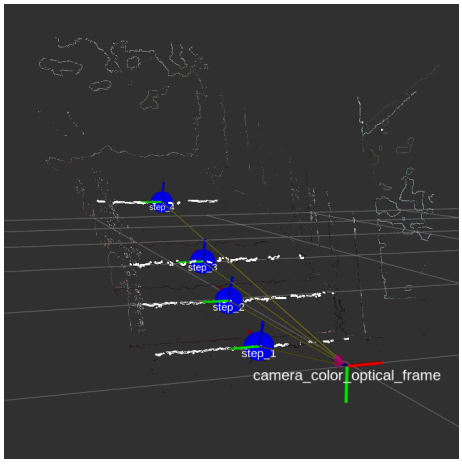
(c) Detecção e filtragem das linhas.



(d) Ajuste do plano que passa pelos pontos das linhas selecionadas.

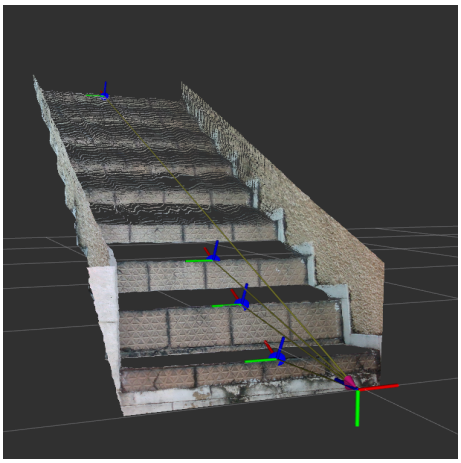


(e) Cálculo do centróide de cada linha pertencente ao plano detectado.

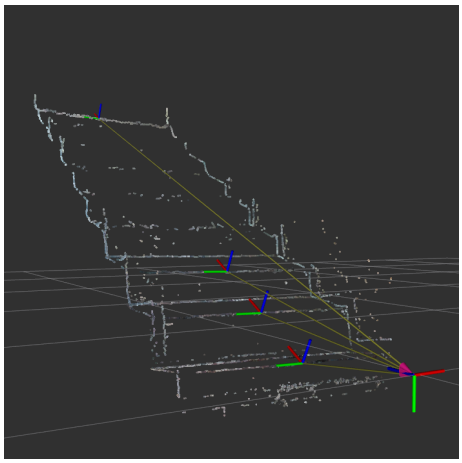


(f) Definição dos respectivos sistemas de coordenadas.

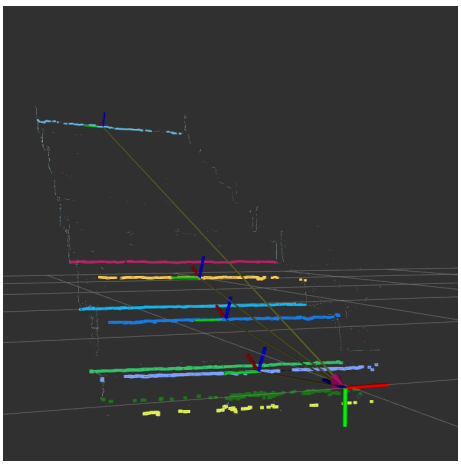
Figura 2.14: Resultado da aplicação dos filtros propostos para detecção de uma escada indoor.



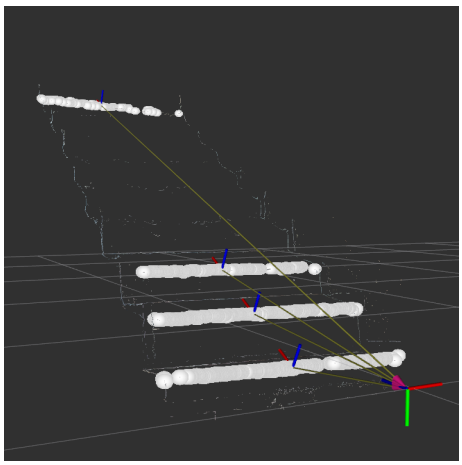
(a) Nuvem de pontos original.



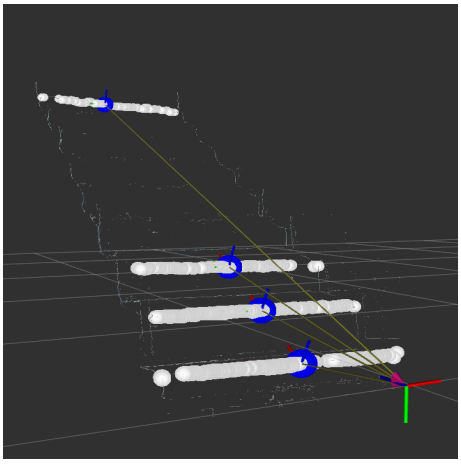
(b) Detecção de bordas.



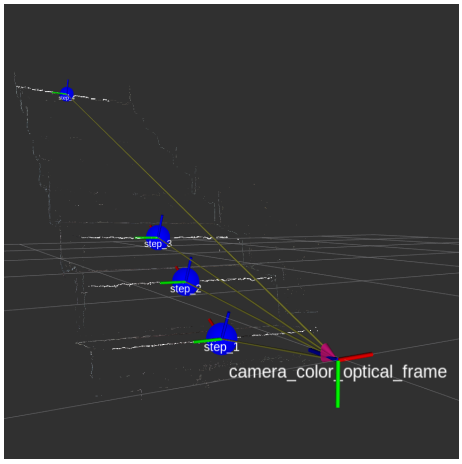
(c) Detecção e filtragem das linhas.



(d) Ajuste do plano que passa pelos pontos das linhas selecionadas.



(e) Cálculo do centróide de cada linha pertencente ao plano detectado.



(f) Definição dos respectivos sistemas de coordenadas.

Figura 2.15: Resultado da aplicação dos filtros propostos para detecção de uma escada outdoor, mais extensa e íngreme, com maior número de pontos oclusos.

às respectivas figuras, ficando claro que ainda que bordas e linhas sejam detectadas de forma correta, nem sempre os pontos estão distribuídos de maneira uniforme. Portanto, medidas absolutas de distância e posição não são confiáveis.

2.5 Conclusões

Neste capítulo foi apresentada uma abordagem para detecção e modelagem de escadas, baseada em sensores 3D com cores. O algoritmo mostra-se robusto em posicionar os sistemas de coordenadas nos locais desejados e orientados de forma coerente, além de fornecer estimativas adicionais com um erro aceitável. Resultados experimentais com desempenho satisfatório foram obtidos. Trabalhos futuros incluem diminuir o custo computacional do pacote desenvolvido, além de agregar funcionalidades de localização, uma vez que se os sistemas de coordenadas ao meio dos degraus é invariante ao longo das iterações, a matriz de transformação homogênea do robô para a escada é conhecida, disponibilizando a posição e orientação do robô para outros fins, como navegação até o início da escada sem necessidade de mais sensores, por exemplo.

Capítulo 3

Controle dos braços durante a transposição de escadas

Como mencionado anteriormente no texto, neste capítulo propõe-se uma metodologia para controlar braços dianteiros e traseiros do robô durante a subida e descida de escadas. A escada é simplificada por um plano inclinado e as forças normais de interação no ponto de contato entre esteira e degrau é considerada suficiente para impulsionar o robô sem que ocorram escorregamentos. O robô é considerado equipado com um sensor capaz de fornecer uma estimativa da orientação e a única premissa sobre conhecimento do terreno é o vetor normal ao plano composto pelas bordas dos degraus. Uma análise da cinemática diferencial do problema é feita, um controlador é proposto e ao final são apresentados resultados experimentais e conclusão.

Considere o AATV apresentado na figura 3.1, com um meio de locomoção principal e equipado com dois pares de braços sobre esteiras, de modo que um par está posicionado à frente, e o outro na traseira do veículo. Os braços são ligados ao corpo por juntas de revolução capazes de imediatamente convergir para uma dada velocidade de referência. Seja $\{0\}$ um sistema de coordenadas fixo, $\{i\}$ com $i = 1,\ldots,4$ sistemas de coordenadas ligados a cada uma das juntas do corpo e $\{r\}$ um sistema de coordenadas posicionado no centroide do robô. Nestes sistemas de coordenadas, setas vermelhas denotam a direção $\vec{x}$ e as azuis a direção $\vec{z}$. Além disso, sejam h_2 e h_3 as respectivas alturas dos sistemas de coordenadas 2 e 3 com respeito ao plano diretamente abaixo.

Na figura 3.1, pode-se notar que uma vez havendo o contato entre a ponta de um braço (sistemas de coordenadas 1 e 4) e um plano, apenas uma condição é necessária para que a respectiva esteira se adeque àquele plano. Uma propriedade básica da geometria é: para que uma reta pertença a um plano, dois pontos na reta devem pertencer ao plano. Portanto, uma abordagem para que um braço, como o traseiro se adeque ao plano diretamente abaixo, dado que o sistema de coordenadas 1 está em contato com este mesmo plano, consiste em fazer com que o sistema de coordenadas 2 esteja a uma altura h_2 igual ao raio da roda. Obviamente, o mesmo se aplica aos braços dianteiros. Logo, uma estratégia de

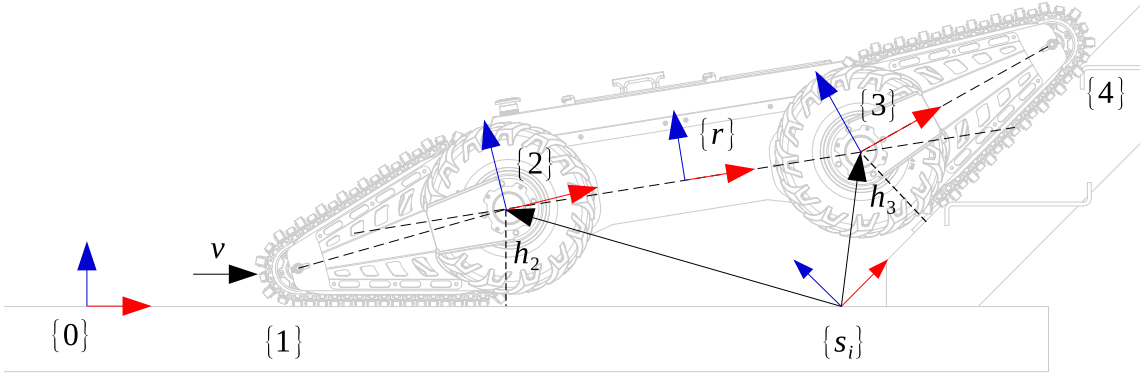


Figura 3.1: A figura representa a transição em sua primeira etapa i.e. 2 está sobre o plano anterior a escada.

controle pode ser formalizada como um problema de regulação das alturas h_2 e h_3 numa dada altura h_{ref} atuando nas velocidades das juntas 2 e 3, i.e. $\dot{\theta}_2$ e $\dot{\theta}_3$ respectivamente.

Uma maneira intuitiva de modelar o sistema consiste em simplesmente interpretar o robô apresentado na figura 3.1 como o manipulador de cadeia aberta com restrição de posição da figura 3.2. De acordo com a fórmula de Gruebler [24] o número de graus de liberdade efetivo é

$$F = 3\,(N - g) + \sum_{i=0}^{g} f_i = 3, \tag{3.1}$$

onde $N = 3$ é o número de elos, $g = 4$ o número de juntas e f_i o número de DoF da i-ésima junta, com $f_i = \{2, 1, 1, 2\}$. Uma abordagem adequada para manipuladores com restrição e número de DoF efetivo tal que o jacobiano das juntas ativas é inversível, é reescrever a velocidade angular das juntas passivas como combinação linear das ativas [16].

Desta forma, considere a formulação do problema como segue. Para um robô como o da figura 3.1, é desejada uma estratégia de controle que gerencie os braços de forma autônoma durante a transposição de escadas, dado que as únicas variáveis conhecidas são: (1) a orientação do robô, (2) a posição angular das juntas e (3) o pitch da escada.

O método proposto divide o processo de transpor uma escada em duas etapas. A primeira, referida no texto como transição no andar inferior é definida como todo momento em que o robô está na iminência de contato, ou em contato com dois planos, um sendo o do andar inferior e outro sendo o plano da escada. A outra etapa é referida no texto como transição no andar superior, e de forma similar, é definida como todo momento em que o robô está na iminência de contato, ou em contato com o plano do andar superior e o plano da escada simultaneamente.

Na seção à seguir são apresentadas a análise da cinemática diferencial e a modelagem do sistema na situação em que o robô se encontra com braços apoiados em dois planos diferentes. Posteriormente uma lei de controle geral é apresentada junto a resultados

experimentais e conclusão.

3.1 Cinemática diferencial

Embora exemplificado em uma situação ideal, pode-se notar que durante transição entre dois planos, o robô da figura 3.1 pode ser genericamente interpretado como o manipulador da figura 3.2. A primeira junta tem dois DoF, uma vez que ativamente avança com velocidade v e rotaciona com velocidade $\dot{\theta}_1$. Segunda e terceira são juntas ativas de revolução, girando com velocidades $\dot{\theta}_2$ e $\dot{\theta}_3$ respectivamente. A última junta é passiva e tem outros dois DoF, uma vez que está restrita a se mover sobre o plano e rotacionar passivamente com velocidade $\dot{\theta}_4$. O número de juntas atuadas nos deixa dá a possibilidade de um operador definir uma velocidade de avanço enquanto a malha fechada atua posicionando os braços de forma apropriada.

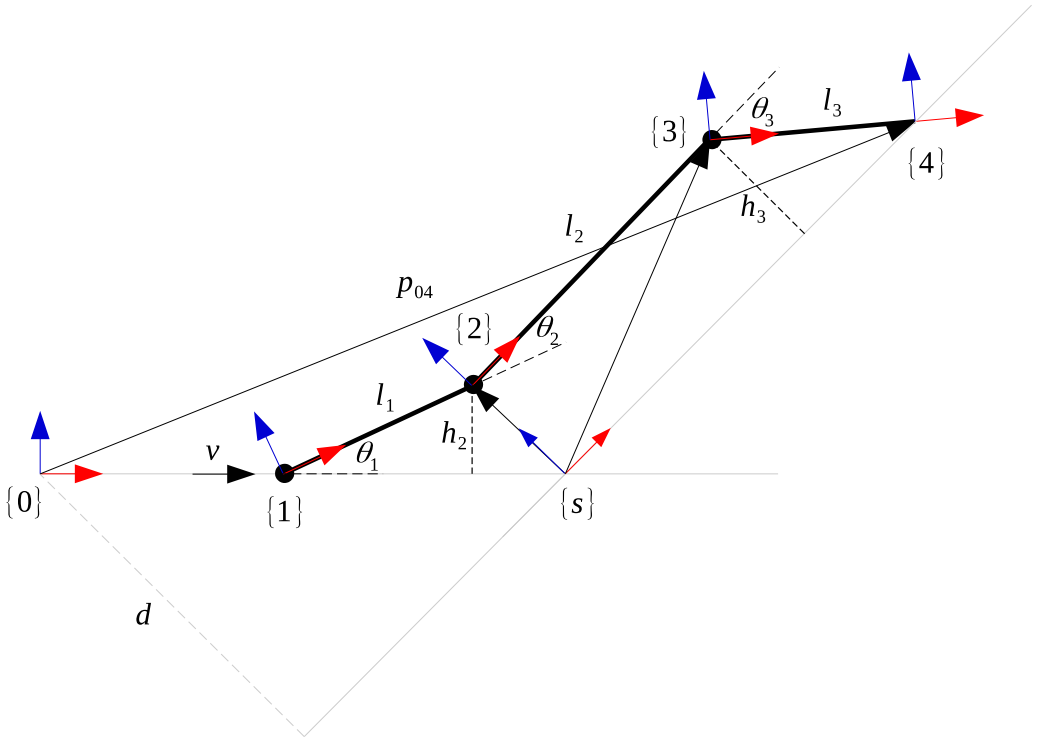


Figura 3.2: Aproximação do robô por um manipulador.

Considere a cadeia cinemática apresentada na figura 3.2 com braços apoiados por dois planos genéricos diferentes. Sejam z_{p_2} e z_{p_3} vetores normais aos planos imediatamente abaixo dos sistemas de coordenadas 2 e 3 respectivamente, e x_{a_1} a direção de avanço do sistema de coordenadas 1. Além disso, defina-se p_{ij} a posição de j relativa à i. Com base nisso, note que o sistema de coordenadas 4 está sujeito a uma restrição do tipo

$$z_{p_3}^T \, p_{04} - d = 0, \tag{3.2}$$

onde, seguindo a cadeia cinemática

$$p_{04} = p_{01} + l_1 R_{01} \vec{x} + l_2 R_{02} \vec{x} + l_3 R_{03} \vec{x}. \tag{3.3}$$

Sabe-se que a derivada de (3.2) no tempo é

$$z_{p_3}^T \dot{p}_{04} = 0, \tag{3.4}$$

onde

$$\dot{p}_{04} = \dot{p}_{01} + \hat{y} p_{12} \dot{\theta}_1 + \hat{y} p_{23} (\dot{\theta}_1 + \dot{\theta}_2) + \hat{y} p_{34} (\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3) \tag{3.5}$$

e $\hat{y} = y \times$ é a matriz anti-simétrica definida anteriormente no texto.

Entretanto

$$\dot{p}_{01} = v x_{a_1}, \tag{3.6}$$

e ao substituir (3.6) e (3.5) em (3.4), é possivel escrever

$$z_{p_3}^T \left(v x_{a_1} + \hat{y} p_{12} \dot{\theta}_1 + \hat{y} p_{23} (\dot{\theta}_1 + \dot{\theta}_2) + \hat{y} p_{34} (\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3) \right) = 0. \tag{3.7}$$

Definam-se as juntas ativas e passivas como

$$\dot{\theta}_p = \dot{\theta}_1 \tag{3.8}$$

e

$$\dot{\theta}_a = \begin{bmatrix} \dot{\theta}_2 & \dot{\theta}_3 \end{bmatrix}^T. \tag{3.9}$$

Logo a equação (3.7) pode ser reescrita da seguinte forma

$$z_{p_3}^T \left(v x_0 + \hat{y} \left(p_{12} + p_{23} + p_{34} \right) \dot{\theta}_p + \hat{y} \begin{bmatrix} p_{23} + p_{34} & p_{34} \end{bmatrix} \dot{\theta}_a \right) = 0, \tag{3.10}$$

e manipulada para chegar à relação

$$\dot{\theta}_p = -\frac{z_{p_3}^T}{z_{p_3}^T J_p} \left(v x_0 + J_a \dot{\theta}_a \right) \tag{3.11}$$

onde $J_p \in \mathbb{R}^{3 \times 1}$ é

$$J_p = \hat{y} \left(p_{12} + p_{23} + p_{34} \right) \tag{3.12}$$

e $J_a \in \mathbb{R}^{3 \times 2}$ é

$$J_a = \hat{y} \begin{bmatrix} p_{23} + p_{34} & p_{34} \end{bmatrix}. \tag{3.13}$$

3.1.1 Modelo das alturas

Como visto anteriormente no texto, a proposta para transição suave entre os diferentes planos na locomoção sobre escadas baseia-se no controle em malha fechada das alturas dos sistemas de coordenadas 2 e 3 com relação ao plano diretamente abaixo. Na figura 3.2, seja a altura h_2 definida como

$$h_2 = z_{p_2}^T \left(p_{02} - p_{0s} \right) \tag{3.14}$$

e sua derivada no tempo

$$\dot{h}_2 = z_{p_2}^T \dot{p}_{02}. \tag{3.15}$$

Sabe-se que

$$\dot{p}_{02} = v\, x_{a_1} + \hat{y}\, p_{12}\, \dot{\theta}_1. \tag{3.16}$$

Substituindo (3.8), (3.11) e (3.16) em (3.15), é possível concluir que

$$\dot{h}_2 = -z_{p_2}^T \hat{y}\, p_{12} \frac{z_{p_3}^T}{z_{p_3}^T J_p} \left(v\, x_{a_1} + J_a\, \dot{\theta}_a \right). \tag{3.17}$$

De forma análoga, para h_3 pode-se seguir o mesmo caminho. Seja h_3 definida como

$$h_3 = z_{p_3}^T \left(p_{03} - p_{0s} \right) \tag{3.18}$$

e sua derivada

$$\dot{h}_3 = z_{p_3}^T \dot{p}_{03}. \tag{3.19}$$

Sabe-se que

$$\dot{p}_{03} = v\, x_{a_1} + \hat{y}\, p_{12}\, \dot{\theta}_1 + \hat{y}\, p_{23} \left(\dot{\theta}_1 + \dot{\theta}_2 \right) \tag{3.20}$$

e novamente substituindo (3.8), (3.11) e (3.20) em (3.19), chega-se em

$$\begin{aligned} \dot{h}_3 = z_{p_3}^T \Bigg(& v\, x_{a_1} - \hat{y} \left(p_{12} + p_{23} \right) \frac{z_{p_3}^T}{z_{p_3}^T J_p} \left(v\, x_{a_1} + J_a\, \dot{\theta}_a \right) \\ & + \hat{y} \begin{bmatrix} p_{23} & 0_{3\times 1} \end{bmatrix} \dot{\theta}_a \Bigg). \end{aligned} \tag{3.21}$$

Note que as equações (3.17) e (3.21) são as velocidades dos respectivos sistemas de coordenadas. Após alguma manipulação é possível separar os temos que estão relacionados à velocidade de avanço do robô v dos termos relativos à rotação das juntas ativas $\dot{\theta}_a$. Portanto, a equação (3.17) pode ser reescrita como

$$\dot{h}_2 = v_2 + J_2^T\, \dot{\theta}_a \tag{3.22}$$

onde

$$v_2 = -v\, z_{p_2}^T\, \hat{y}\, p_{12}\, \frac{z_{p_3}^T}{z_{p_3}^T J_p}\, x_{a_1}, \tag{3.23}$$

$$J_2^T = -z_{p_2}^T\, \hat{y}\, p_{12}\, \frac{z_{p_3}^T}{z_{p_3}^T J_p}\, J_a, \tag{3.24}$$

e a equação (3.21) como

$$\dot{h}_3 = v_3 + J_3^T\, \dot{\theta}_a \tag{3.25}$$

onde

$$v_3 = v\, z_{p_3}^T \left(I_{3\times 3} - \hat{y}\, \left(p_{12} + p_{23}\right)\, \frac{z_{p_3}^T}{z_{p_3}^T J_p} \right) x_{a_1} \tag{3.26}$$

e

$$J_3^T = z_{p_3}^T\, \hat{y} \left(\begin{bmatrix} p_{23} & 0_{3\times 1} \end{bmatrix} - \left(p_{12} + p_{23}\right)\, \frac{z_{p_3}^T}{z_{p_3}^T J_p}\, J_a \right). \tag{3.27}$$

Portanto, com (3.22) e (3.25) pode-se reescrever o sistema como

$$\dot{H} = V(\theta, v) + J(\theta)\, \dot{\theta}_a \tag{3.28}$$

onde

$$H = \begin{bmatrix} h_2 & h_3 \end{bmatrix}^T, \tag{3.29}$$

$$V(\theta, v) = \begin{bmatrix} v_2 & v_3 \end{bmatrix}^T, \tag{3.30}$$

e

$$J(\theta) = \begin{bmatrix} J_2^T \\ J_3^T \end{bmatrix}. \tag{3.31}$$

3.2 Controle para gerenciamento dos braços

Considere o sistema (3.28). Seja $H_r \in \mathbb{R}^2$ uma altura de referência constante e o erro de altura e definido como

$$e = H - H_r. \tag{3.32}$$

Sua derivada portanto é

$$\dot{e} = \dot{H},$$

e logo

$$\dot{e} = V(\theta, v) + J(\theta)\, \dot{\theta}_a. \tag{3.33}$$

Deseja-se uma lei de controle por realimentação de estados de forma que a origem do sistema em malha fechada seja assintoticamente estável sob as seguinte hipóteses: (**A1**) o vetor de juntas ativas é a entrada do sistema, i.e. $\dot{\theta}_a = u$; (**A2**) o termo $V(\theta, v)$ é limitado;

(**A3**) a matriz $J(\theta)$ possui posto completo.

A lei de controle proposta é dada pela equação

$$u = -J^{-1}\left(V + k_p\, e + k_i \int_0^t e(\tau)\, d\tau\right), \tag{3.34}$$

onde $k_p, k_i \in \mathbb{R}^{2\times 2}$ são matrizes positivas definidas.

Teorema 1. *Considere o sistema (3.33), a lei de controle (3.34) e as hipóteses (**A1–A3**). A origem do sistema em malha fechada é globalmente exponencialmente estável.*

Prova 1. *Em malha fechada, o termo integral proporciona uma dinâmica de segunda ordem*

$$\ddot{e} + k_p\dot{e} + k_i e = 0. \tag{3.35}$$

Deste modo, a dinâmica linear do erro (3.35) é globalmente exponencialmente estável.

3.3 A máquina de estados para as transições

Durante as transições, a tarefa principal dos braços é puxar todo o corpo na direção da superfície desejada. Enquanto ativo, o controlador proposto atua nos braços promovendo a regulação em malha fechada das alturas h_2 e h_3 num valor h_{ref}, igual ao raio das rodas r_w, de forma que ambas as esteiras se adequem às suas respectivas superfícies. Tanto as referidas alturas quanto a direção de avanço do sistema de coordenadas 1 são funções dos vetores normal e tangente do plano imediatamente abaixo. Como mencionado anteriormente no texto, apesar do uso das posições cartesianas na cinemática diferencial, medidas de posição não são necessárias. Assumindo o conhecimento do pitch do robô θ_r, medido por uma IMU, da posição angular das juntas θ_2 e θ_3 e do pitch da escada θ_s, as alturas podem ser estimadas apenas seguindo a cadeia cinemática.

O gerenciamento do método de cálculo das alturas, o Jacobiano e as direções de projeção necessárias na equação (3.34) é feito por uma máquina de estados finitos. Entretanto, antes de apresentá-lo, na figura 3.3, seja r_g o raio da engrenagem posicionada na ponta dos braços.

3.3.1 Máquina de estados na transição do andar inferior

No andar inferior, considere as variáveis para se mover ao longo da cadeia cinemática

$$\theta_1 = \theta_r - \theta_2\,,$$

$$\theta_4 = \theta_1 + \theta_2 + \theta_3 + \theta_s\,.$$

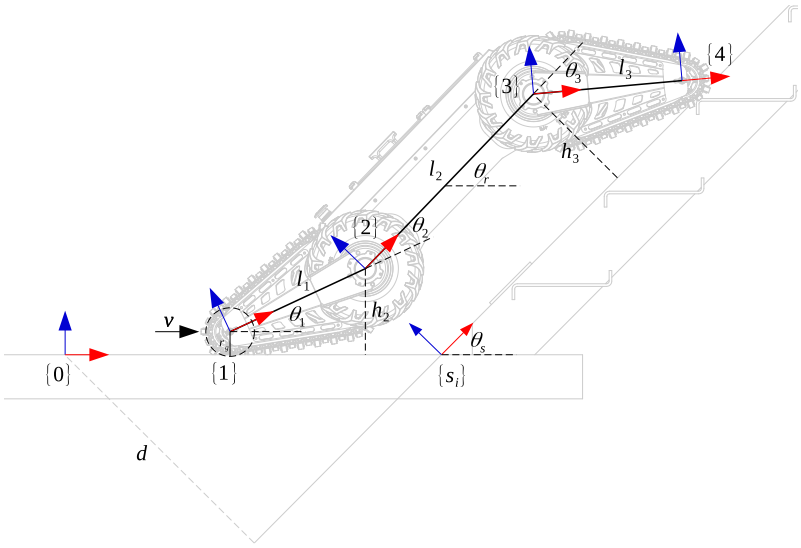


Figura 3.3: Aproximação do robô por um manipulador.

As direções e forma para cálculo das alturas usadas na equação (3.34) são apresentadas na tabela 3.1.

State	z_{p_2}	z_{p_3}	x_{a_1}	h_2	h_3
A_{inf}	z_0	z_{s_i}	x_0	$l_1\sin(\theta_1)+r_g$	$-l_3\sin(\theta_4)+r_g$
B_{inf}	z_{s_i}	z_{s_i}	x_0	$-(l_2\sin(\theta_4+\theta_3)+l_3\sin(\theta_4)+r_g)$	$-l_3\sin(\theta_4)+r_g$

Tabela 3.1: Tabela com as direções de projeção e cálculo das alturas na máquina de estados para o andar inferior.

A transição no andar inferior é dividida em quatro estados da figura 3.4 que dependem da posição das juntas 1, 2 e 3 com relação ao início da escada s_i. Considere o robô num estado inicial 0_{inf}. Como na figura 3.5, neste estado, supõe-se o robô alinhado à escada, afastado da mesma por uma distância tal que os braços dianteiros estejam na iminência de tocar a borda do primeiro degrau. Enquanto em 0_{inf}, a máquina de estados pode evoluir apenas para o estado A_{inf} por ação direta do operador.

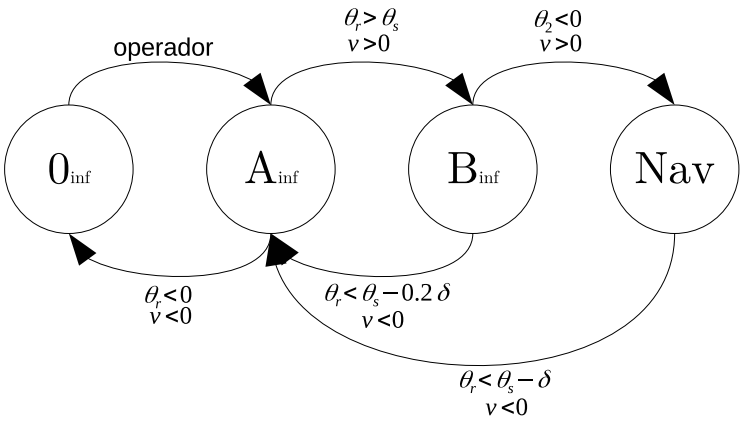


Figura 3.4: Máquina de estados finitos na transição de entrada da escada.

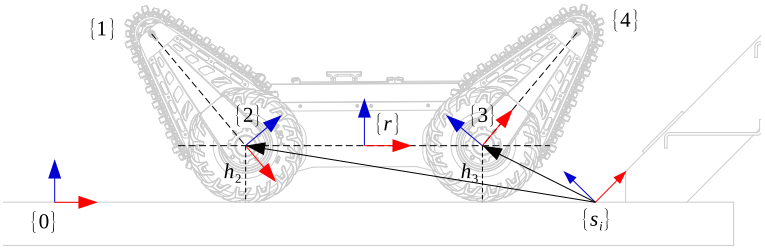


Figura 3.5: No estado 0 o robô se encontra sobre o piso inferior e alinhado à escada, ligeiramente afastado do primeiro degrau, pronto para iniciar o processo de subida.

Em A_{inf}, na figura 3.6, uma malha fechada de controle mantém os braços dianteiros adequados ao plano da escada e os traseiros ao plano do andar inferior. Neste estado, a máquina de estados pode evoluir para B_{inf} uma vez que o pitch do robô torne-se superior ao da escada i.e. $\theta_r > \theta_s$ com $v > 0$, ou para 0_{inf} caso o pitch do robô se iguale ou torne-se menor ao do andar inferior i.e. $\theta_r \leq 0$ com $v < 0$.

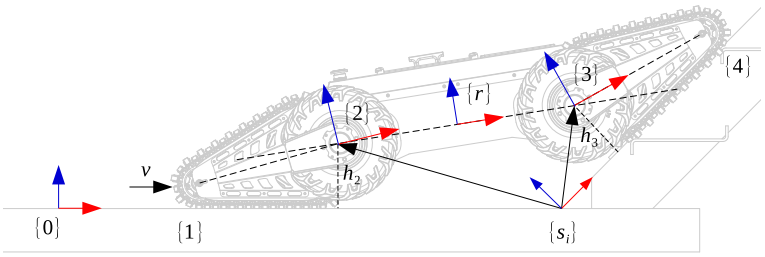


Figura 3.6: No estado A_{inf} os *flippers* dianteiros encontram-se apoiados na borda dos degraus, já sobre o plano da escada, enquanto os traseiros encontram-se no plano do piso inferior. Na figura, o robô está posicionado de forma ideal, com controle em malha fechada movendo os braços para que sigam em conformidade com os respectivos planos à medida que avança.

Em B_{inf}, na figura 3.7, embora os braços traseiros mantenham-se apoiados sobre o plano do andar inferior, a origem do sistema de coordenadas 2 está sobre o plano da escada. Portanto a altura h_2 no caso é calculada com relação ao plano da escada, de forma que o robô possa deslocar-se mantendo sua inclinação igual à da escada até a próxima transição de estado. Enquanto em B_{inf} a máquina de estados pode evoluir tanto para Nav assim que ambos os braços estejam posicionados sobre um plano com $\theta_2 = -\theta_3$ com $v > 0$, quanto para A_{inf} quando o pitch do robô torna-se menor que o da escada $\theta_r < \theta_s - \delta$ (onde δ é uma pequena constante para adiantar ou atrasar a transição de estados) com $v < 0$.

No estado Nav, na figura 3.8, o robô encontra-se navegando inteiramente sobre a escada. Tendo em vista o pressuposto de homogeneidade nas dimensões dos degraus, em tese, não há necessidade de controle malha fechada para h_2 e h_3, visto que como as alturas

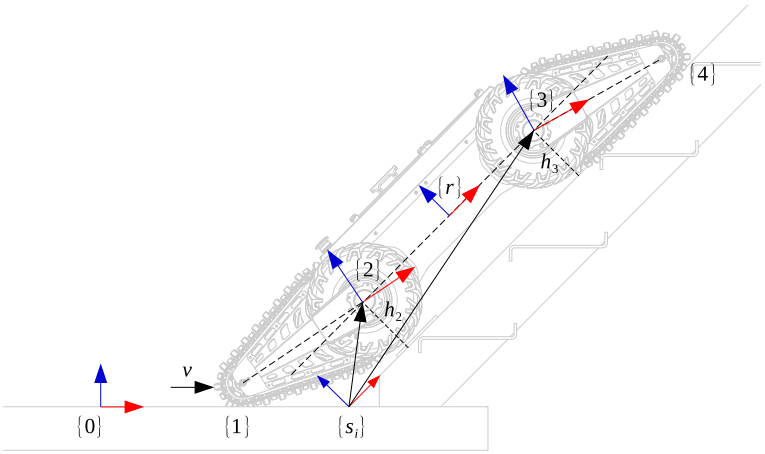


Figura 3.7: No estado B_{inf}, embora o sistema de coordenadas 2 já se encontre sobre o plano da escada, note que o apoio do braço traseiro permanece sobre o plano do piso inferior. À medida que avança ambos os braços movem-se de forma automática, mantendo a inclinação do coincidente com a da escada, suavizando toda a etapa de transição para qualquer dos dois estados possíveis.

não variam com o avanço, basta posicionar os braços de forma fixa, adequados ao plano da escada. Como visto anteriormente no texto, ao navegar sobre a escada pequenas variações no rumo podem aproximar o robô das laterais da escada. Portanto, a preocupação neste estado é controlar o rumo de forma a manter o robô em direção ao fim da escada. A literatura apresenta diversas abordagens que variam desde simples controladores proporcionais como em [27], [13] o LQR proposto em [10] e outras estratégias para seguimento de caminho.

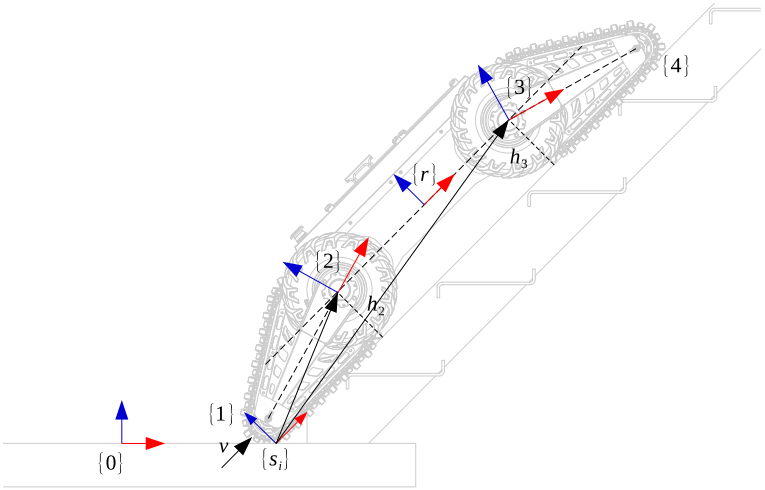


Figura 3.8: No estado Nav, já com as esteiras sobre uma superfície assumida plana, o avanço deixa de causar alterações nas altura h_2 e h_3. Restando apenas uma rotina para manter o robô afastado das extremidades da escada durante a navegação, discutida mais afrente no texto.

3.3.2 Máquina de estados na transição do andar superior

Nesta transição, os braços traseiros encontram-se apoiados no plano da escada enquanto os dianteiros, no plano do andar superior. Portanto, os ângulos da primeira e última juntas são redefinidos como

$$\theta_1 = \theta_r - \theta_2 + \theta_s \,,$$

e

$$\theta_4 = \theta_1 + \theta_2 + \theta_3 - \theta_s \,.$$

Na figura 3.9, no andar superior perceba que a junta 1 se move sobre o plano da escada com direção de avanço x_{s_i} enquanto o efetuador 4 está restrito a se movimentar sobre o plano do andar superior. De forma análoga à anterior, no piso superior há uma subdivisão

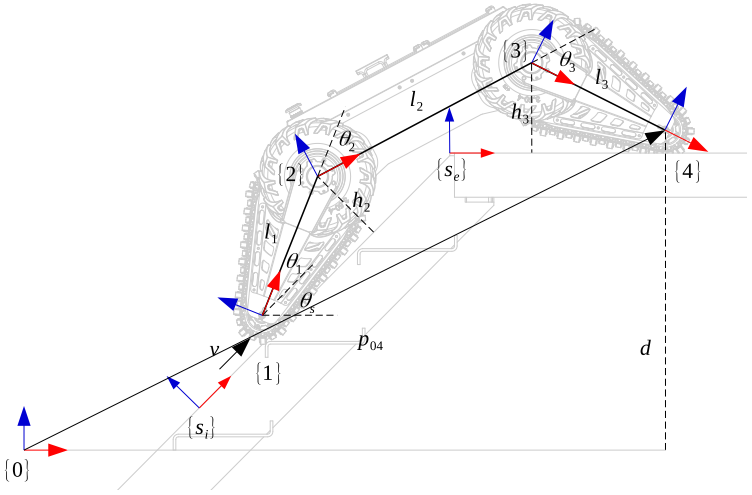


Figura 3.9: Situação que correlaciona a distancia de 2 até o piso superior, h', e a altura h_3 à um breve apoio do casco sobre a borda do degrau.

em outras quatro estados da figura 3.10, que neste caso, além de dependerem da posição das juntas 1, 2 e 3 com relação ao final da escada s_e, dependem também da condição de segurança adicional mencionada mais à frente no texto. As direções e forma para cálculo das alturas usadas na equação (3.34) são apresentadas na tabela 3.2.

State	z_{p_2}	z_{p_3}	x_{a_1}	h_2	h_3
A_{sup}	z_{s_i}	z_0	x_{s_i}	$l_1 \sin(\theta_1) + r_g$	$-l_3 \sin(\theta_4) + r_g$
B_{sup}	z_0	z_0	x_{s_i}	$-(l_2 \sin(\theta_4 + \theta_3) + l_3 \sin(\theta_4) + r_g)$	$-l_3 \sin(\theta_4) + r_g$

Tabela 3.2: Tabela com as direções de projeção e cálculo das alturas na máquina de estados para o andar superior.

No estado Nav, na figura 3.11, o robô encontra-se alinhado ao espelho dos degraus, posicionado sobre a linha média entre as extremidades laterais, com o sistema de coordenadas 2 sobre o plano da escada e 3 arbitrariamente próximo de s_e. Nesta transição, o robô eventualmente encontra uma situação onde os braços dianteiros perdem o contato

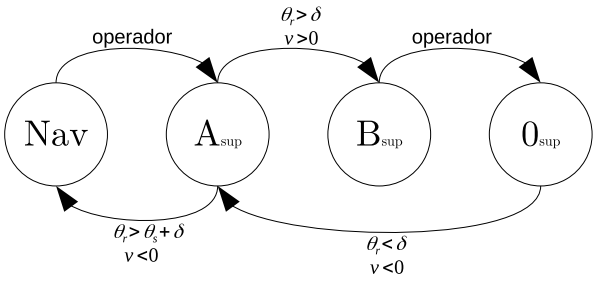


Figura 3.10: Máquina de estados finitos na transição de saída da escada.

com a superfície do andar superior. A transição para o estado A_{sup} se dá do plano da escada em direção ao andar superior e ocorre por uma ação do operador.

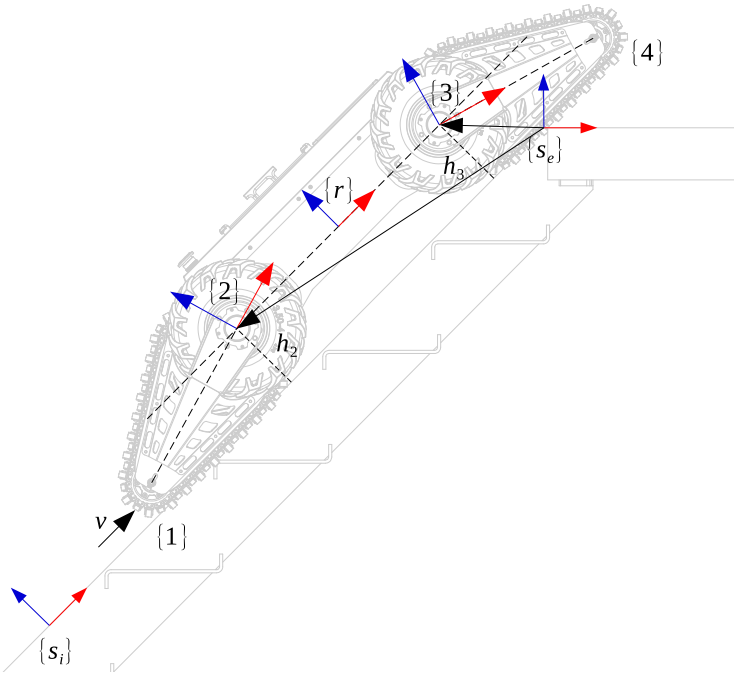


Figura 3.11: Máquina de estados finitos na transição no piso superior.

No estado A_{sup}, na figura 3.12, uma malha fechada de controle mantém os braços dianteiros adequados ao plano do andar superior e os traseiros sobre o plano da escada, modificando a inclinação do robô até que esta convirja para a mesma do andar superior no caso da subida, ou do plano da escada na descida. Enquanto neste estado, a máquina de estados pode evoluir para B_{sup} uma vez que o pitch do robô coincida ou torne-se maior que do andar superior i.e. $\theta_r \geq 0$ com $v > 0$, ou para Nav caso coincida com o da escada $\theta_r > \theta_s + \delta$ com $v < 0$.

A figura 3.13 evidencia uma situação onde o robô se apoia em um ponto no último degrau da escada. Seja l_2 o comprimento total do segundo elo, l_{22} o comprimento do segundo elo antes do ponto de contato e l_{23} o comprimento restante após o ponto de contato. Seja d_h a distância entre a linha média do robô e seu fundo, θ_s a inclinação da

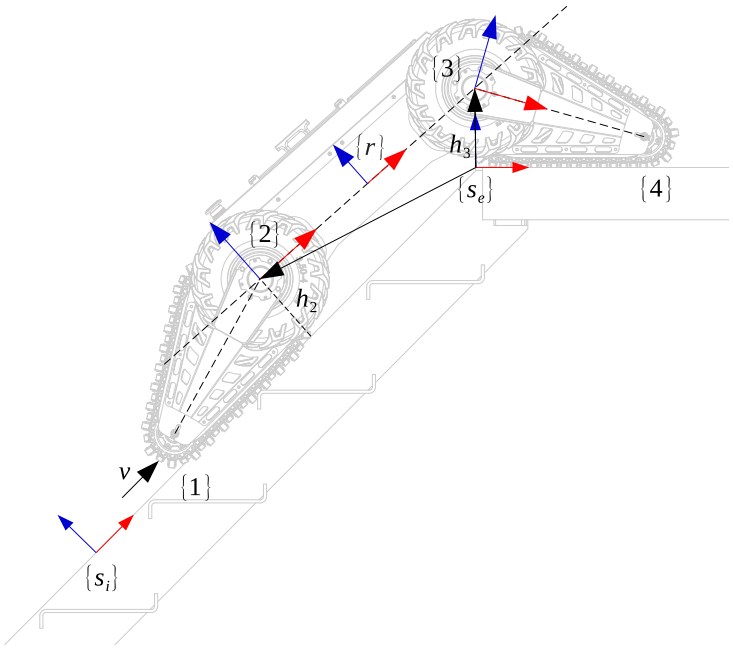


Figura 3.12: Máquina de estados finitos na transição de saída da escada.

escada e θ_r a do robô. Definam-se as distâncias auxiliares

$$h'_2 = h_2 - \frac{d_h}{\cos(\theta_s - \theta_r)} \tag{3.36}$$

e

$$h'_3 = h_3 - \frac{d_h}{\cos(\theta_r)}. \tag{3.37}$$

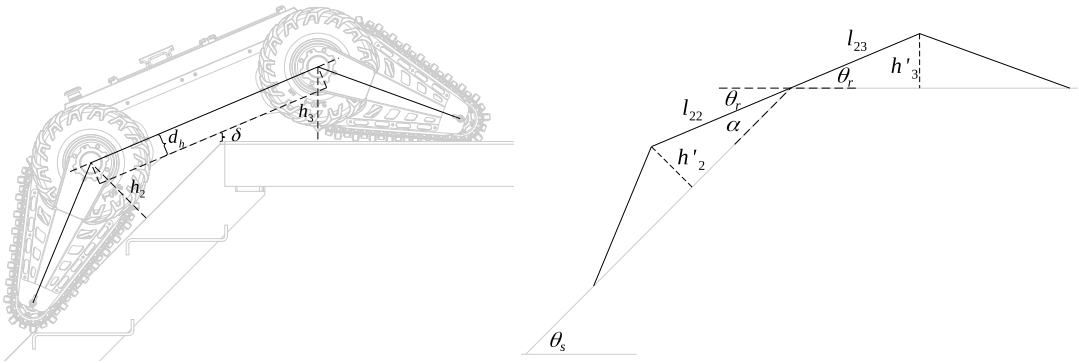


Figura 3.13: Uma distância δ pequena caracteriza eminência de colisão entre o fundo do robô e a borda do último degrau. Na figura esquemática à direita é possível estabelecer uma relação trigonométrica que deve ser satisfeita para que o valor δ seja sempre positivo, evitando-se portanto a colisão.

Sabe-se que

$$l_2 = l_{22} + l_{23} \tag{3.38}$$

onde

$$l_{22} = \frac{h'_2}{\sin(\theta_s - \theta_r)} \tag{3.39}$$

e

$$l_{23} = \frac{h'_3}{\sin\theta_r}. \tag{3.40}$$

Substituindo (3.39) e (3.40) em (3.38) tem-se

$$l_2 = \frac{h'_2}{\sin(\theta_s - \theta_r)} + \frac{h'_3}{\sin\theta_r}. \tag{3.41}$$

Escrevendo h_2 em função de h_3 com finalidade de determinar o valor de altura para que o contato ocorra, chega-se em

$$h_2 = \left(l_2 - \frac{h'_3}{\sin\theta_r}\right)\sin(\theta_s - \theta_r) + \frac{d_h}{\cos(\theta_s - \theta_r)}. \tag{3.42}$$

Portanto, é possível escolher uma combinação de h_2 e h_3 tal que o corpo mantenha-se acima da borda, livre de colisões. Desta forma, no estado A_{sup} mantém-se a regulação de ambas as alturas em malha fechada num valor igual r_s, como na abordagem anterior, adicionando a condição

$$h_{2_{ref}} > \left(l_2 - \frac{h'_3}{\sin\theta_r}\right)\sin(\theta_s - \theta_r) + \frac{d_h}{\cos(\theta_s - \theta_r)} \tag{3.43}$$

que deve ser respeitada durante todo o processo. Na proximidade de desrespeitar a condição, incrementa-se $h_{2_{ref}}$ por um pequeno valor Δ, levantando a traseira e afastado o fundo da colisão.

Em B_{sup}, figura 3.14, movendo-se na direção do andar superior, quando ná iminência da transição entre a junta 2 e o andar superior, há a necessidade de ajustar os braços mantendo o robô na inclinação equivalente à do andar, garantindo que traseira não levante devido o movimento de avanço. Este estado tem duração relativamente curta, visto que o robô está praticamente fora da escada, tendo o papel apenas de afastar o robô de forma segura. Em B_{sup} a máquina de estados pode apenas evoluir 0_{sup} por intervenção do operador, completando a subida. Uma vez em 0_{sup} o robô pode inciar a decida posicionando os braços levemente abaixados, avançando lentamente em direção à escada até que começe a inclinar com $\theta_r < 0$.

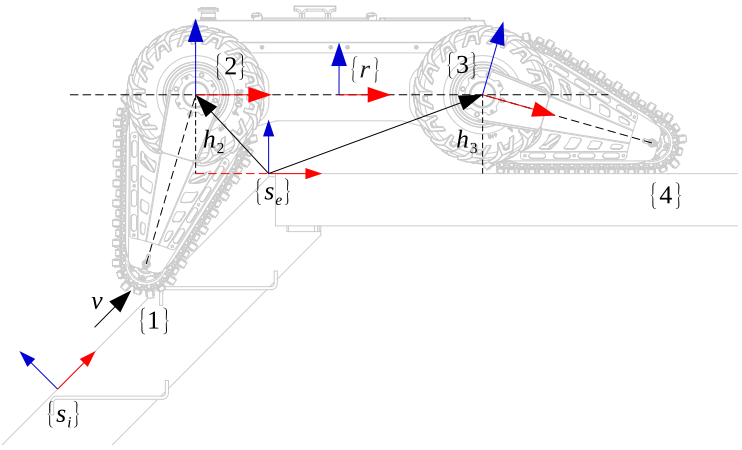


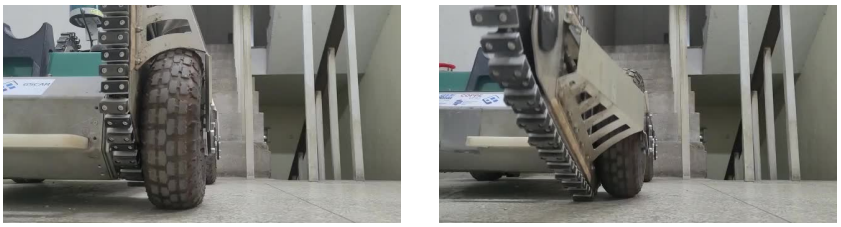
Figura 3.14: Máquina de estados finitos na transição de saída da escada.

3.4 Resultados experimentais

Para a análise de desempenho do método proposto, o algoritmo foi implementado para o robô Rosi, apresentado na introdução. Durante os experimentos, a orientação do robô foi obtida através da IMU embarcada e os experimentos foram conduzidos em duas escadas com diferentes níveis de dificuldade. No experimento, o robô foi deixado sobre rodas, no estado 0, alinhado às escadas e afastado do primeiro degrau por uma distância de aproximadamente 20cm. Com um comando externo, o robô inicia seu avanço em direção ao andar superior com velocidade constante e braços controlados de forma autônoma através da lei de controle (3.34) tanto na subida quanto descida. Como mencionado anteriormente no texto, a convergência do erro de alturas para zero implica que ambas as esteiras estão adequadas à suas respectivas superfícies imediatamente abaixo. Desta forma, serão apresentados os gráficos de convergência dos erros de altura junto ao gráfico do pitch do veículo, que eventualmente ao final de cada etapa deverá convergir para o pitch da escada ou para zero.

As figuras 3.15 e 3.16 mostram sequências de imagens e convergência do erro de altura durante a subida da primeira escada. Relativamente curta com apenas 2.1m, possui inclinação de aproximadamente 30°, piso e espelho de medidas iguais à 300mm e 170mm, respectivamente. Sua menor inclinação diminui a componente do peso na direção tangente ao plano da escada, proporcionando melhores condições de tração e diminuindo escorregamento e perda de pontos suporte.

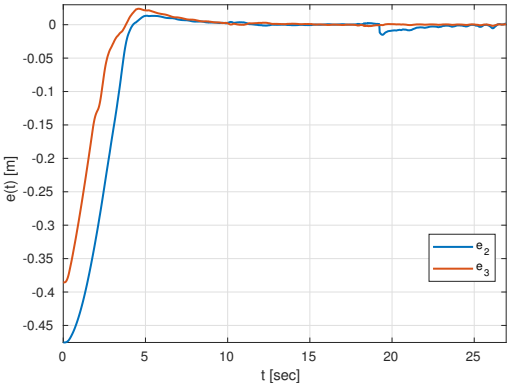
Já a segunda escada, com 4.1m de comprimento, inclinação de aproximadamente 40°, piso e espelho de medidas iguais à 230mm e 180mm, esta escada apresenta maiores complicações. Além da maior inclinação, a borda dos degraus onde se apoiam as esteiras tem superfícies mais lisas e com maior raio de curvatura, prejudicando a tração, tornando-a mais difícil que a primeira. Em função de seu maior comprimento, a escada apresentada nesta seção permite uma melhor compreensão das manobras realizadas pelos braços tanto



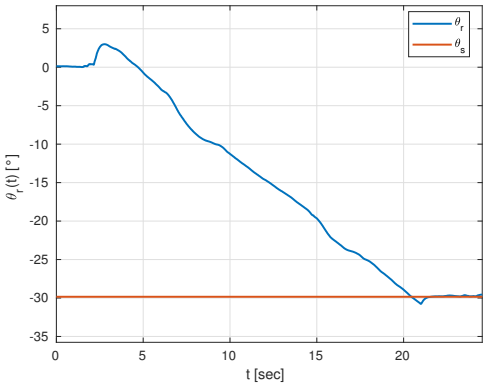
(a) (b)

(c) (d)

(e) (f)



(g) Erro de altura



(h) Convergência do pitch

Figura 3.15: A sequência de imagens durante a transição para plano da escada, começando no andar inferior (a) e terminando sobre a escada (f). Os gráficos mostram a convergência do erro de altura e pitch do robô apesar das perturbações como escorregamento e perda dos pontos de suporte. É possível notar um comportamento quase linear na convergência do pitch do robô para o mesmo da escada.

na subida quanto descida. As figuras 3.17 e 3.18 apresentam os resultados para a subida enquanto 3.20 e 3.19 para a descida.

Nos resultados apresentados para os erros de alturas, podem-se notar pequenos sobressaltos que se derivam da máquina de estados proposta. Na metodologia apresentada, tanto o cálculo das alturas quanto as direções de projeção na lei de controle variam em

(a)

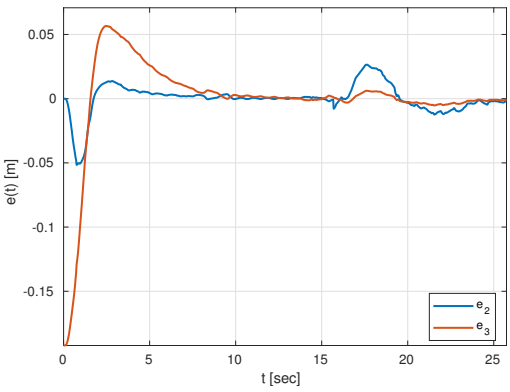
(b)

(c)

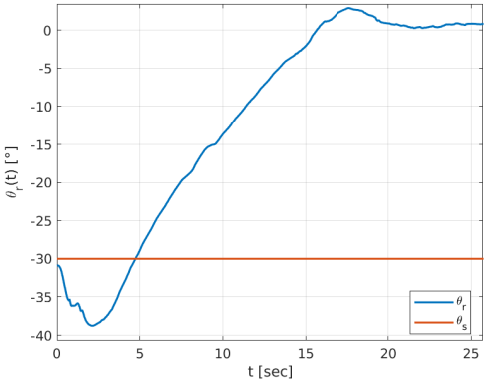
(d)

(e)

(f)



(g) Erro de altura



(h) Convergência do pitch

Figura 3.16: Ao final da escada, na iminência da chegada ao andar superior (a) os braços dianteiros estão dispostos como se ainda estivessem sobre o plano da escada. A falta de contato entre a ponta dos braços e o plano viola a hipótese feita para o cálculo das alturas h_2 e h_3, causando uma resposta peculiar nos erro de altura em um primeiro momento. Entretanto, o método proposto se prova eficaz em manter os erros de altura em zero e a convergência do pitch de forma suave.

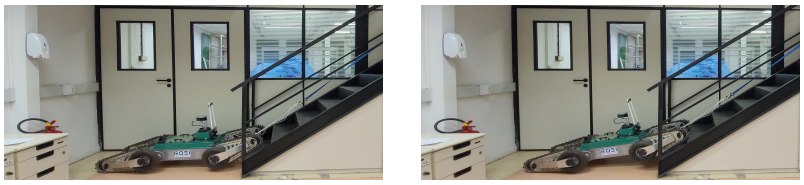
função do estado atual da máquina de estados, de forma que durante os eventos de transição, uma pequena perturbação é esperada. Entretanto, seus efeitos se mostram pequenos, sem comprometer a transposição de escadas, seja na subida ou descida. Já nos resultados

para convergência do pitch para os valores apropriados, a metodologia proposta se mostra eficaz em todos os resultados apresentados com exceção da figura 3.19. Nesta etapa, com braços traseiros apoiados sobre o plano da escada e dianteiros sobre o andar superior, a regulação das alturas nos valores discutidos anteriormente faz com que o pitch do robô supere o da escada por um fator limitante δ, usado como evento de transição de estados, sinalizando que deve-se ir para o estado de navegação.

3.5 Conclusões

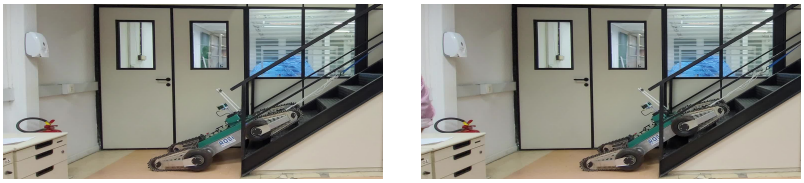
Neste capítulo foi apresentada uma nova estratégia para modelar um AATV durante o processo de subida de escadas. Uma lei de controle por realimentação de estados foi proposta para gerenciamento automático dos braços com propriedades de estabilidade e convergência do erro de altura para zero. Resultados experimentais validam tanto a modelagem proposta quanto a lei de controle. Usando esta metodologia, o robô mostrou-se capaz de continuamente ajustar os braços às respectivas superfícies diretamente abaixo. Apesar de pequenos desvios causados pela diferença no método de cálculo das alturas em diferentes estados, o termo integral foi capaz de manter o erro de altura em zero apesar do movimento de avanço constante, escorregamento das esteiras e perda de pontos suporte. Para transições ainda mais suaves, melhorar a qualidade da estimativa das alturas é um ponto a melhorar. Apesar da abordagem ter sido desenvolvida num contexto de subida de escadas, ele pode ser generalizado para transposição de outros obstáculos com mínimas mudanças.

(a) (b)

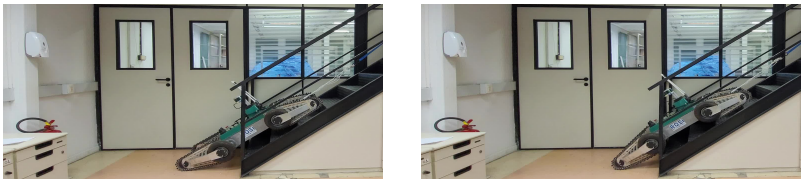


(c) (d)

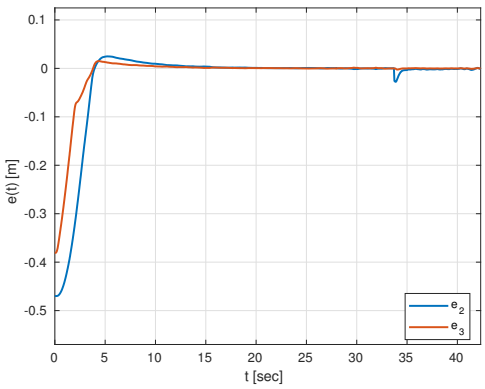
(e) (f)



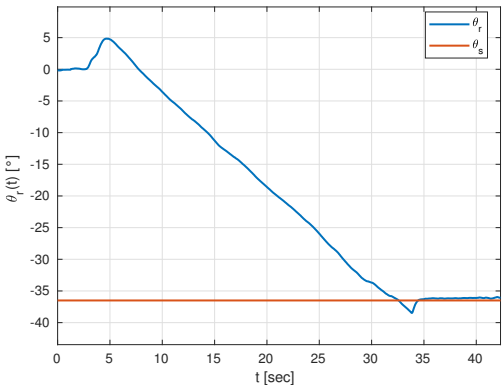
(g) (h)



(i) (j)



(k) Erro de altura.



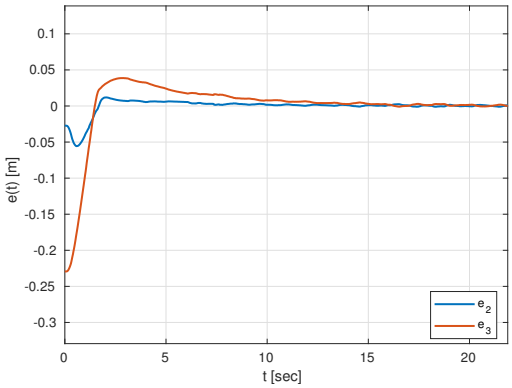
(l) Convergência do pitch.

Figura 3.17: Convergência do erro de alturas e pitch durante a transição do andar inferior para o plano da escada.

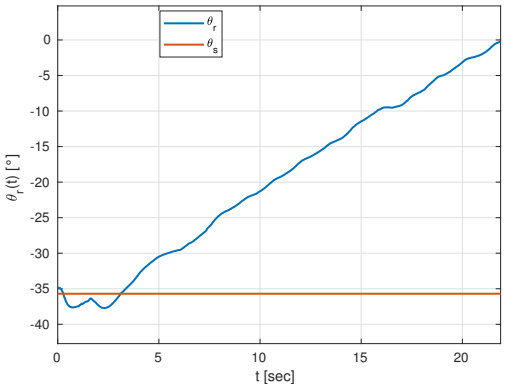
(a) (b)

(c) (d)

(e) (f)



(g) Erro de altura.



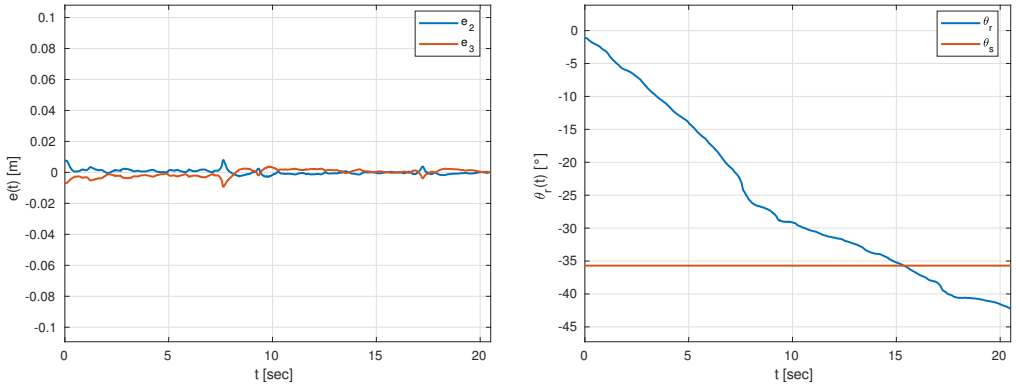
(h) Convergência do pitch.

Figura 3.18: Convergência do erro de alturas e pitch durante a transição do plano da escada para o andar superior. Neste caso, diante da impossibilidade de sair completamente da escada por falta de espaço, o robô apenas se mantém-se parcialmente sobre o andar superior.

(a) (b)

(c) (d)

(e) (f)



(g) Erro de altura. (h) Convergência do pitch.

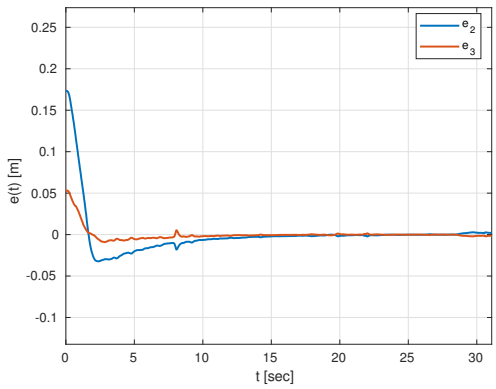
Figura 3.19: Convergência do erro de alturas e pitch durante a transição do andar superior para o plano da escada. A convergência do pitch do robô não acontece em um primeiro momento. Maiores detalhes serão apresentados ao final desta seção.

(a) (b)

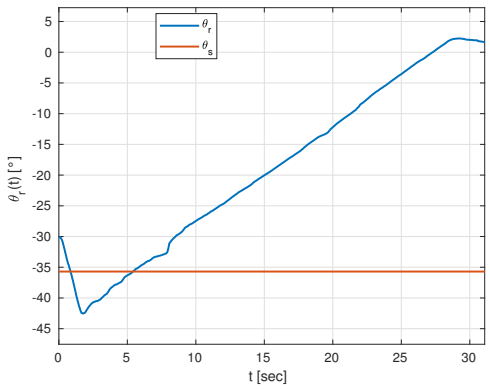
(c) (d)

(e) (f)

(g) (h)



(i) Erro de altura.



(j) Convergência do pitch.

Figura 3.20: Convergência do erro de alturas e pitch durante a transição do plano da escada para o andar inferior, finalizando todo o processo de subida e descida.

Capítulo 4

Controle para seguimento de caminho

A estratégia para seguimento de caminho proposta é baseada em [35], onde o caminho é entendido como a curva de nível formada na interseção entre duas funções objetivo Φ_1 e Φ_2, como na Seção 3.2 deste texto. Aqui o caminho S é interpretado com um segmento de curva no plano XY de um sistema de coordenadas qualquer. Desta forma, o caminho pode ser definido como $S = \{(x,y) \,|\, \Phi(x,y) = 0\}$ onde $\Phi(x,y)$ é a função cuja curva de nível quando $z = 0$ é o caminho desejado. Uma estratégia para fazer com que um robô siga um caminho consiste em fazer com que sua posição p satisfazer uma restrição do tipo $\Phi(p) = 0$, uma vez que para todo $p \notin S$, $\Phi(p) \neq 0$.

Na figura 4.1, considere um robô cinemático modelado pelo sistema de equações diferenciais

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos(\phi) & 0 \\ \sin(\phi) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{4.1}$$

onde (x,y) é a posição cartesiana do robô no plano, ϕ é o ângulo de rumo e v e ω são respectivamente as velocidades linear e angular. Portanto, seja $p_{0r} = \begin{bmatrix} x & y \end{bmatrix}^T$ a posição cartesiana do robô no plano XY Para este robô, é desejado uma estratégia de controle que leve uma função objetivo $\Phi(p_{0r})$ para zero, fazendo-o seguir o caminho na correta direção de movimentação.

Seja

$$\Phi : \mathbb{R}^2 \to \mathbb{R}^n \tag{4.2}$$

uma função real de classe C^1, multivalorada ou escalar. Sabe-se que sua derivada no tempo é

$$\dot{\Phi} = \nabla_{\Phi}^T \dot{p}_{0r} \tag{4.3}$$

onde

$$\nabla_{\Phi}^T = \begin{bmatrix} \frac{d\Phi}{dx} & \frac{d\Phi}{dy} \end{bmatrix} \tag{4.4}$$

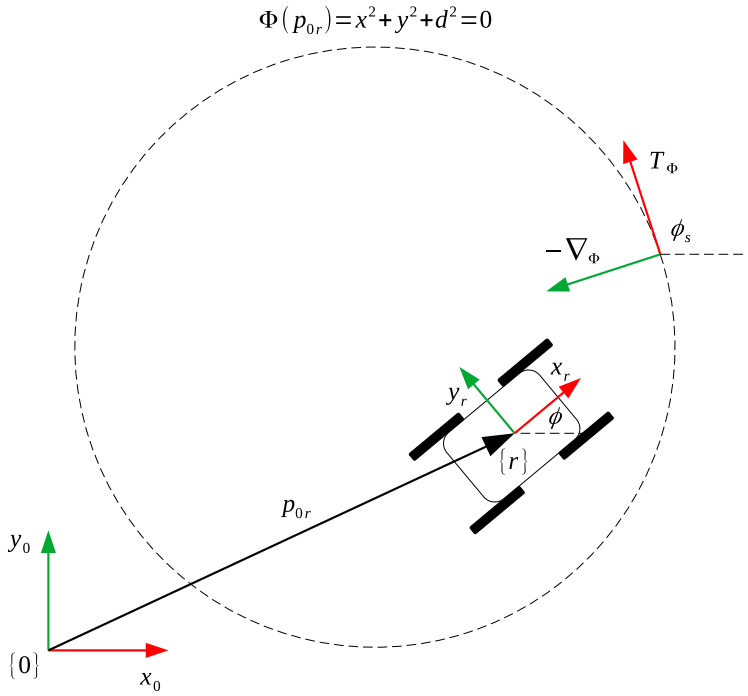


Figura 4.1: Na figura, o caminho desejado é a linha tracejada, formada pela curva de nível quando $\Phi(p_{0r}) = x^2 + y^2 - d^2 = 0$. Na posição retratada, $\Phi(p_{0r}) \neq 0$ e o controlador está empurrando o robô para o caminho desejado.

e

$$\dot{p}_{0r} = v\, x_r.$$

Note que no plano XY, a equação $\nabla_\Phi^T T_\Phi = 0$ é suficiente para o a determinação do vetor tangente

$$T_\Phi^T = \begin{bmatrix} \frac{d\Phi}{dy} & -\frac{d\Phi}{dx} \end{bmatrix} \tag{4.5}$$

e do ângulo entre o gradiente e o eixo x_0

$$\phi_s = \tan^{-1}\left(\frac{T_{\Phi_y}}{T_{\Phi_x}}\right). \tag{4.6}$$

A equação (4.3) pode ser reescrita pela definição do produto interno entre vetores como

$$\dot{\Phi} = v\, \|\nabla_\Phi\|\, \|x_r\| \cos\left(\phi - \left(\phi_s + \frac{\pi}{2}\right)\right). \tag{4.7}$$

Deste modo, considere o erro de orientação

$$e_\phi = \phi - \phi_s. \tag{4.8}$$

Substituindo (4.8) em (4.7) e sabendo que $\|x_r\| = 1$ pode-se reescrever (4.7) como

$$\dot{\Phi} = v \, \|\nabla_\Phi\| \, \sin(e_\phi). \tag{4.9}$$

Da equação (4.8), tem-se que a a derivada do erro de orientação no tempo é

$$\dot{e_\phi} = \dot{\phi} - \dot{\phi_s}, \tag{4.10}$$

onde da equação (4.1) $\dot{\phi} = \omega$ e $\dot{\phi_s}$ pode ser obtida analiticamente diferenciando a equação (4.6) com relação ao tempo. Sabe-se que

$$\dot{\phi_s} = \frac{1}{1 + \left(\frac{T_{\Phi_y}}{T_{\Phi_x}}\right)^2} \frac{d}{dt}\left(\frac{T_{\Phi_y}}{T_{\Phi_x}}\right) \tag{4.11}$$

e

$$\frac{d}{dt}\left(\frac{T_{\Phi_y}}{T_{\Phi_x}}\right) = \frac{\frac{dT_{\Phi_y}}{dt} T_{\Phi_x} - T_{\Phi_y} \frac{dT_{\Phi_x}}{dt}}{T_{\Phi_x}^2}. \tag{4.12}$$

Entretanto, note que

$$\frac{dT_{\Phi_x}}{dt} = \frac{d}{dt}\frac{d\Phi}{dy} = \frac{d}{dy}\frac{d\Phi}{dt} = \left(\frac{d\nabla\Phi}{dy}\right)^T \dot{p}_{0r} \tag{4.13}$$

e

$$\frac{dT_{\Phi_y}}{dt} = -\frac{d}{dt}\frac{d\Phi}{dx} = -\frac{d}{dx}\frac{d\Phi}{dt} = -\left(\frac{d\nabla\Phi}{dx}\right)^T \dot{p}_{0r}, \tag{4.14}$$

onde

$$\left(\frac{d\nabla\Phi}{dy}\right)^T = \begin{bmatrix} \frac{d}{dy}\frac{d\Phi}{dx} & \frac{d^2\Phi}{dy^2} \end{bmatrix} \tag{4.15}$$

e

$$\left(\frac{d\nabla\Phi}{dx}\right)^T = \begin{bmatrix} \frac{d^2\Phi}{dx^2} & \frac{d}{dx}\frac{d\Phi}{dy} \end{bmatrix}. \tag{4.16}$$

Substituindo as equações (4.13) e (4.14) em (4.12), é possível concluir que

$$\frac{d}{dt}\left(\frac{T_{\Phi_y}}{T_{\Phi_x}}\right) = -\dot{p}_{0r}^T \begin{bmatrix} \frac{d\nabla\Phi}{dx} & \frac{d\nabla\Phi}{dy} \end{bmatrix} \begin{bmatrix} T_{\Phi_x} \\ T_{\Phi_y} \end{bmatrix} \frac{1}{T_{\Phi_x}^2} = -\frac{\dot{p}_{0r}^T H_\Phi T_\Phi}{T_{\Phi_x}^2}, \tag{4.17}$$

onde H_Φ é a matriz Hessina de Φ, definida como

$$H_\Phi = \begin{bmatrix} \frac{d\nabla\Phi}{dx} & \frac{d\nabla\Phi}{dy} \end{bmatrix}. \tag{4.18}$$

Deste modo, substituindo (4.17) em (4.11), chega-se a conclusão que

$$\dot{\phi}_s = -\frac{\dot{p}_{0r}^T H_\Phi T_\Phi}{\|T_\Phi\|^2}. \tag{4.19}$$

Portanto, por (4.9), (4.19) e (4.1), pode-se finalmente reescrever a dinâmica do sistema como

$$\begin{aligned} \dot{\Phi} &= v \|\nabla_\Phi\| \sin(e_\phi) \\ \dot{e}_\phi &= \frac{\dot{p}_{0r}^T H_\Phi T_\Phi}{\|T_\Phi\|^2} + \omega \end{aligned} \quad , \tag{4.20}$$

de forma que a função objetivo Φ pode ser entendida como uma distância entre o robô e o caminho desejado e e_ϕ o erro de orientação entre o robô e a tangente do caminho.

4.1 Controle para seguimento de caminho

O objetivo do controlador proposto é fazer com que o robô siga um caminho S, como definido no início do capítulo, usando a velocidade angular do robô, ω, como a entrada. Isto é: achar uma lei de controle por realimentação de estados, tal que o sistema (4.20) em malha fechada tenha garantidas propriedades de estabilidade e convergência dos estados (Φ, e_ϕ) para zero. Deste modo, considere as seguintes premissas:

- (**A1**) A velocidade de avanço $v(t)$ é uma função conhecida tal que $|v(t)| > 0$;
- (**A2**) A função objetivo Φ é tal que $\|\nabla_\Phi\| \neq 0 \; \forall \Phi \neq 0$;
- (**A3**) A entrada u é a velocidade angular do robô i.e $u = \omega$.

A lei de controle proposta é dada pela seguinte equação

$$u = -\frac{\dot{p}_{0r}^T H_\Phi T_\Phi}{\|T_\Phi\|^2} - k_g \, v \, \Phi \, \|\nabla_\Phi\| \, \frac{\sin\left(e_\phi\right)}{e_\phi} - k_p \, e_\phi \tag{4.21}$$

onde $\left(k_g, k_p\right)$ são constantes positivas. A análise de estabilidade é feita a seguir.

Teorema 2. *Considere o sistema (4.20), as premissas (**A1-A3**) e a lei de controle proposta em (4.21). A origem do sistema em malha fechada é assintoticamente estável para a região* $D = \left\{\left(\Phi, e_\phi\right) \in \mathbb{R}^2\right\}$.

Prova 2. *Considere a seguinte função de Lyapunov*

$$2V = k_g \, \Phi^2 + e_\phi^2$$

positiva definida em todo o espaço de estados. Sua derivada no tempo é dada por

$$\dot{V} = k_g \, \Phi \, \dot{\Phi} + e_\phi \, \dot{e}_\phi.$$

Portanto, segue que

$$\dot{V} = k_g \, v \, \Phi \, \|\nabla_\Phi\| \, \sin(e_\phi) + e_\phi \, u. \tag{4.22}$$

Ao substituir as equações (4.21) em (4.22) chega-se a conclusão que

$$\dot{V} = -k_p e_\phi^2 \leq 0 \tag{4.23}$$

em toda a região D.

Note que o sistema em malha fechada é

$$\begin{aligned} \dot{\Phi} &= v \, \|\nabla_\Phi\| \, \sin(e_\phi) \\ \dot{e}_\phi &= -k_g \, v \, \Phi \, \|\nabla_\Phi\| \, \frac{\sin(e_\phi)}{e_\phi} - k_p \, e_\phi \end{aligned} \quad . \tag{4.24}$$

Uma vez que V é continuamente diferenciável, positiva definida, radialmente ilimitada e $\dot{V} \leq 0$ ao longo de toda a região em análise, pelo Princípio da Invariância de LaSalle [20], todas as trajetórias convergem para o maior conjunto invariante $\bar{\Omega}$ em $\Omega = \left\{\left(\Phi, e_\phi\right) : \dot{V} = 0\right\} = \left\{\left(\Phi, e_\phi\right) : e_\phi = 0\right\}$. Entretanto, em Ω tem-se que $e_\phi = 0 = \dot{\Phi}$. Além disso, para que V se mantenha em Ω é necessário que $\dot{e}_\phi = 0$. Uma vez que k_g é uma constante positiva, e que de acordo com as premissas, $v(t)$ é diferente de zero, $\frac{\sin(e_\phi)}{e_\phi} = 1$ quando $e_\phi \to 0$ e $\|\nabla_\Phi\| \neq 0$ para $\Phi \neq 0$, tem-se que $\Phi = 0$. Portanto, o único ponto no conjunto $\bar{\Omega}$ é a origem $\left\{\left(\Phi, e_\phi\right) = 0\right\}$, que é globalmente assintoticamente estável. Desta forma, para toda condição inicial $\left\{\left(\Phi(t_0), e_\phi(t_0)\right) \in D\right\}$ as variáveis de estado eventualmente convergem para $(0, 0)$.

4.1.1 Resultados preliminares

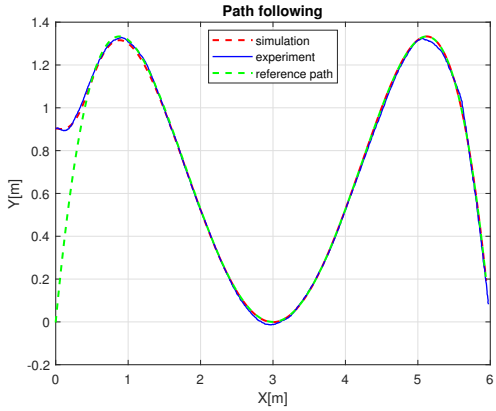
Para validar a estratégia proposta para seguimento de caminho, a lei de controle foi implementada e resultados experimentais foram gerados com um robô móvel sobre esteiras chamado Diane, predecessor da Rosi. A posição e rumo foram ambos estimados através de um sensor laser Velodyne VLP-16 e do pacote Hector Slam implementado em ROS.

Uma função polinomial de quarto grau foi selecionada como caminho, de forma que a função objetivo é $f : \mathbb{R}^2 \to \mathbb{R}$

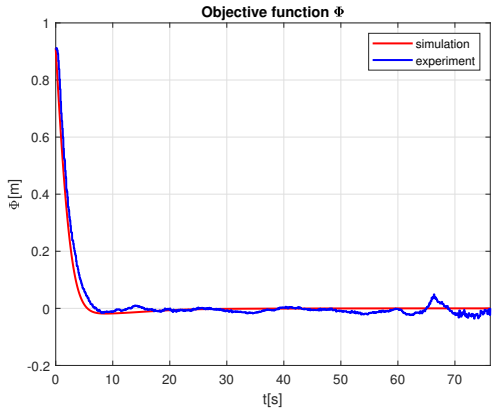
$$\Phi(x, y) = y - f(x) = y - \sum_{k=1}^{4} a_k \, x^k,$$

onde $a_1 = 3.5556$, $a_2 = -2.9630$, $a_3 = 0.7901$ e $a_4 = -0.0658$. Note que a primeira e

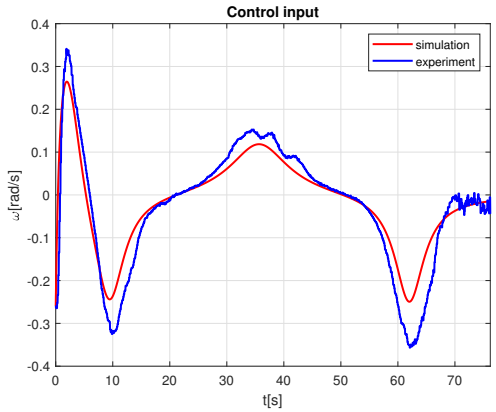
segunda derivadas de f em função de x podem ser calculadas de forma analítica. As simulações foram feitas usando o sistema de equações diferenciais em (4.20). Nesta seção, resultados experimentais e simulados são apresentados juntos no mesmo gráfico na figura 4.2.



(a)



(b)



(c)

Figura 4.2: Resultados para o controlador proposto.

4.2 Parametrização do caminho para subida de escadas

Como o objetivo é manter o robô posicionado ao centro da escada, o caminho mais simples para a implementação consiste em uma reta do tipo $ax + by + c = 0$. Na figura 4.3, o caminho mais adequado é composto pela reta que intercepta as origens de ambos os sistemas de coordenadas $\{s_i\}$ e $\{s_e\}$. Note ainda, que o caminho em questão pode ser parametrizado pela equação

$$\Phi(x, y) = y, \tag{4.25}$$

uma vez que basta apenas manter-se sobre o eixo x_{s_i}. Caminhos parametrizados por equações como (4.25) respeitam a premissa (**A2**), visto que o gradiente ∇_Φ é vetor unitário e constante. Além disso, o mesmo acontece com o vetor tangente T_Φ, de forma que o termo $\dot{\phi}_s$ da equação (4.20) é nulo.

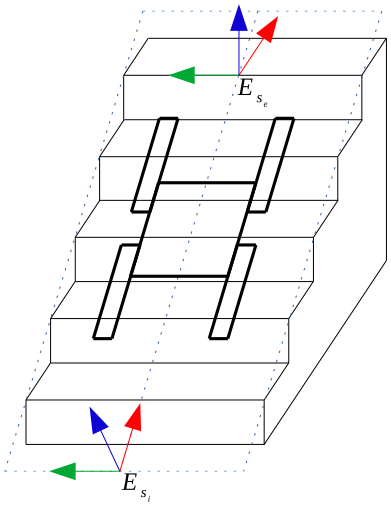


Figura 4.3: O caminho desejado, uma reta entre a origem dos sistemas de coordenadas $\{s_i\}$ e $\{s_e\}$ é parametrizado pela equação $y = 0$.

Capítulo 5

Conclusões gerais

Neste trabalho foram apresentadas metodologias para detecção e gerenciamento automático dos braços durante a transposição de escadas para robôs do tipo AATV equipados apenas com uma IMU e um sensor RGB-D.

A estratégia de gerenciamento dos braços de forma contínua parece ser um ponto faltante na literatura, visto que trabalhos publicados recentemente, ou necessitam de um modelo demasiadamente rico do terreno, ou se assentam em soluções inventivas e empíricas. A solução apresentada nesta dissertação utiliza apenas conceitos bem fundamentos da cinemática de manipuladores junto à um simples controlador PI, somente sendo necessários o início da escada, o fim e o vetor normal ao plano composto pelas bordas. Resultados experimentais validam a abordagem e mostram que o controlador proposto foi capaz de adequar os braços aos respectivos planos e gerenciar as transições de forma autônoma tanto na subida quanto na descida. Apesar da complexidade da interação entre o robô e a escada, as simplificações feitas acerca dos pontos de contato entre as esteiras e os degraus se mostram válidas desde que haja múltiplos pontos de suporte.

A detecção e modelagem de escadas apresentada neste trabalho também mostraram-se eficientes sobretudo ao modelar escadas de dimensões pequenas ou medianas. Numa escada de maiores dimensões, como a segunda dos resultados experimentais, a oclusão de pontos em função da altura baixa dos AATVs e o alto nível de ruído dificultam a identificação dos degraus mais afastados, ainda que a modelagem da escada não seja comprometida. Muito embora este trabalho não tenha evoluído a tal ponto, a detecção dos sistemas de coordenadas posicionados nas bordas dos degraus pode ser aprimorada para que sirva como uma fonte da matriz de transformação homogênea entre o sistema de coordenadas do respectivo AATV e a escada, podendo ser utilizada junto a um EKF para navegação, sem uso de mais sensores.

Ainda que não tenha sido abordado, o controle para navegação sobre escadas é um consenso na literatura de subida de escadas, sendo complementar à estratégia de gerenciamento dos braços. Para AATVs, normalmente o terreno sobre a escada é assumido como um plano que intercepta as bordas dos degraus. Assume-se também a homogeneidade na

distribuição da força de contato entre as esteiras e o plano imediatamente abaixo. Resultados experimentais mostram que a simplificação é válida, embora pequenos distúrbios no rumo ocorram em função da interação entre esteiras e bordas. Ao avançar, variações no rumo, mesmo que corrigidas posteriormente, são invariavelmente responsáveis por aproximar o robô das extremidades laterais da escada.

5.1 Trabalhos futuros

Esta dissertação propôs uma metodologia para gerenciamento dos braços durante a transição entre os planos dos andares superior e inferior e o plano da escada. Entretanto, como visto anteriormente, uma vez sobre a escada o controle do rumo para que o robô se mantenha afastado das bordas é imperativo. Dado que o problema de localização esteja resolvido, o conhecimento prévio dos sistemas de coordenadas ao início e ao fim da escada pode ser usado para parametrização de um caminho entre os dois.

Uma sugestão para trabalhos futuros é a implementação de um controle para seguimento de caminho que mantenha o robô sobre a reta entre os sistemas de coordenadas ao início e fim da escada, equidistante das extremidades e apontado na direção correta de subida durante toda a navegação entre os dois referidos sistemas de coordenadas.

Referências Bibliográficas

[1] Christopher G Atkeson, Benzun P Wisely Babu, Nandan Banerjee, Dmitry Berenson, Christoper P Bove, Xiongyi Cui, Mathew DeDonato, Ruixiang Du, Siyuan Feng, Perry Franklin, et al. No falls, no resets: Reliable humanoid behavior in the darpa robotics challenge. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 623–630. IEEE, 2015.

[2] Dana H Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[3] Filippo Basso, Emanuele Menegatti, and Alberto Pretto. Robust intrinsic and extrinsic calibration of rgb-d cameras. *IEEE Transactions on Robotics*, 34(5):1315–1332, 2018.

[4] Luca Bruzzone and Giuseppe Quaglia. Review article: locomotion systems for ground mobile robots in unstructured environments. *Mechanical Sciences*, 3:49–62, 07 2012.

[5] Widodo Budiharto. Design of tracked robot with remote control for surveillance. In *Proceedings of the 2014 International Conference on Advanced Mechatronic Systems*, pages 342–346. IEEE, 2014.

[6] Horst Bunke. Fast segmentation of range images into planar regions by scan line grouping. *Mach. Vis. Appl.*, 7:115–122, 06 1994.

[7] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[8] Lucas Carvalho. Semi-autonomous locomotion on ladders for mobile robots with tracks. Masters dissertation, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Rio de Janeiro, Brasil, 2016.

[9] Derek S Chan, Rôb Klér Silva, João C Monteiro, and Fernando Lizarralde. Efficient stairway detection and modeling for autonomous robot climbing. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5916–5921. IEEE, 2017.

[10] Yogita Choudhary, Nidhi Malhotra, Pratyush Kumar Sahoo, and Shyam Kamal. Data-driven modeling of a track-based stair-climbing wheelchair. In *2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1000–1005. IEEE, 2021.

[11] Jeffrey A Delmerico, David Baran, Philip David, Julian Ryde, and Jason J Corso. Ascending stairway modeling from dense depth imagery for traversability analysis. In *2013 IEEE International Conference on Robotics and Automation*, pages 2283–2290. IEEE, 2013.

[12] Anna Eilering, Victor Yap, Jeff Johnson, and Kris Hauser. Identifying support surfaces of climbable structures from 3d point clouds. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6226–6231. IEEE, 2014.

[13] Daisuke Endo, Atsushi Watanabe, and Keiji Nagatani. Stair climbing control of 4-degrees-of-freedom tracked vehicle based on internal sensors. In *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 112–117. IEEE, 2016.

[14] Péter Fankhauser, Marko Bjelonic, C Dario Bellicoso, Takahiro Miki, and Marco Hutter. Robust rough-terrain locomotion with a quadrupedal robot. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5761–5768. IEEE, 2018.

[15] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[16] Gustavo Freitas, Antonio Leite, and Fernando Lizarralde. Kinematic control of constrained robotic systems. *Sba Controle - Automacao Sociedade Brasileira de Automatica*, 22:559–572, 12 2011.

[17] Mario Gianni, Federico Ferri, Matteo Menna, and Fiora Pirri. Adaptive robust three-dimensional trajectory tracking for actively articulated tracked vehicles. *Journal of Field Robotics*, 33(7):901–930, 2016.

[18] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Pearson, 2018.

[19] Marco Hutter, Christian Gehring, Andreas Lauber, Fabian Gunther, Carmine Dario Bellicoso, Vassilios Tsounis, Péter Fankhauser, Remo Diethelm, Samuel Bachmann, Michael Blösch, et al. Anymal-toward legged robots for harsh environments. *Advanced Robotics*, 31(17):918–931, 2017.

[20] H.K. Khalil. *Nonlinear Systems*. Prentice Hall, 1996.

[21] Yusuke Kida, Satoshi Kagami, Toru Nakata, Makiko Kouchi, and Hiroshi Mizoguchi. Human finding and body property estimation by using floor segmentation and 3d labelling. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, volume 3, pages 2924–2929. IEEE, 2004.

[22] D. Kim, D. Carballo, J. Di Carlo, B. Katz, G. Bledt, B. Lim, and S. Kim. Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2464–2470, 2020.

[23] Young Hun Lee, Yoon Haeng Lee, Hyunyong Lee, Hansol Kang, Yong Bum Kim, Jun Hyuk Lee, Luong Tin Phan, Sungmoon Jin, Hyungpil Moon, Ja Choon Koo, et al. Whole-body motion and landing force control for quadrupedal stair climbing. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4746–4751. IEEE, 2019.

[24] Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, USA, 1st edition, 2017.

[25] Evgeni Magid, Takashi Tsubouchi, Eiji Koyanagi, and Tomoaki Yoshida. Static balance for rescue robot navigation: Losing balance on purpose within random step environment. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 349–356. IEEE, 2010.

[26] Carlos Marques, João Cristóvão, Pedro Lima, João Frazão, Isabel Ribeiro, and Rodrigo Ventura. Semi-autonomous robot for rescue operations. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. Citeseer, 2006.

[27] Ehsan Mihankhah, Arash Kalantari, Ehsan Aboosaeedan, Hamid D Taghirad, S Ali, and A Moosavian. Autonomous staircase detection and stair climbing for a tracked mobile robot using fuzzy controller. In *2008 IEEE International Conference on Robotics and Biomimetics*, pages 1980–1985. IEEE, 2009.

[28] EZ Moore, D Campbell, Felix Grimminger, and Martin Buehler. Reliable stair climbing in the simple hexapod'rhex'. In *Proceedings 2002 IEEE International Conference on Robotics and Automation*, volume 3, pages 2222–2227. IEEE, 2002.

[29] Anastasios I Mourikis, Nikolas Trawny, Stergios I Roumeliotis, Daniel M Helmick, and Larry Matthies. Autonomous stair climbing for tracked vehicles. *The International Journal of Robotics Research*, 26(7):737–758, 2007.

[30] Stefan Obwald, Jens-Steffen Gutmann, Armin Hornung, and Maren Bennewitz. From 3d point clouds to climbing stairs: A comparison of plane segmentation approaches for humanoids. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 93–98, 2011.

[31] Yoshito Okada, Keiji Nagatani, Kazuya Yoshida, Satoshi Tadokoro, Tomoaki Yoshida, and Eiji Koyanagi. Shared autonomy system for tracked vehicles on rough terrain based on continuous three-dimensional terrain scanning. *Journal of Field Robotics*, 28(6):875–893, 2011.

[32] Unmesh Patil, Aniket Gujarathi, Akshay Kulkarni, Aman Jain, Lokeshkumar Malke, Radhika Tekade, Kartik Paigwar, and Pradyumn Chaturvedi. Deep learning based stair detection and statistical image filtering for autonomous stair climbing. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pages 159–166. IEEE, 2019.

[33] Filipe Rocha, Gabriel Garcia, Raphael FS Pereira, Henrique D Faria, Thales H Silva, Ricardo HR Andrade, Evelyn S Barbosa, André Almeida, Emanuel Cruz, Wagner Andrade, et al. Rosi: A robotic system for harsh outdoor industrial inspection-system design and applications. *Journal of Intelligent & Robotic Systems*, 103(2):1–22, 2021.

[34] Solomon Steplight, Geofrey Egnal, S-H Jung, Daniel B Walker, Camillo J Taylor, and James P Ostrowski. A mode-based sensor fusion approach to robotic stair-climbing. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)*, volume 2, pages 1113–1118. IEEE, 2000.

[35] Jian Wang, Iurii A. Kapitaniuk, Sergey A. Chepinskiy, Dongliang Liu, and Aleksandr J. Krasnov. Geometric path following control in a moving frame. *IFAC-PapersOnLine*, 48(11):150–155, 2015. 1st IFAC Conference onModelling, Identification andControl of Nonlinear SystemsMICNON 2015.

[36] Qun Zhang, Shuzhi Sam Ge, and Pey Yuen Tao. Autonomous stair climbing for mobile tracked robot. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 92–98. IEEE, 2011.