JOSÉ GERALDO DAS NEVES ORLANDI

# SISTEMA DE VISÃO A LASER PARA MAPEAMENTO DE SUPERFÍCIE DE NAVEGAÇÃO DE ROBÔS QUADRÚPEDES

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Doutor em Engenharia Elétrica, na área de concentração em Automação.
Orientador: Prof. Dr. Paulo Faria Santos Amaral.
Co-orientador: Prof. Dr. Antônio Bento Filho.

VITÓRIA
2008

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

---

Orlandi, José Geraldo das Neves, 1966-

O71s Sistema de visão a laser para mapeamento de superfície de navegação de robôs quadrúpedes / José Geraldo das Neves Orlandi. – 2008.

212 f. : il.

Orientador: Paulo Faria Santos Amaral.

Co-Orientador: Antônio Bento Filho.

Tese (doutorado) – Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Robôs móveis. 2. Visão de robô. 3. Navegação de robôs móveis. 4. Processamento de imagens. 5. Desvio de obstáculos . I. Amaral, Paulo Faria Santos. II.Bento Filho, Antônio. III. Universidade Federal do Espírito Santo. Centro Tecnológico. IV. Título.

CDU: 621.3

---

**JOSÉ GERALDO DAS NEVES ORLANDI**

# SISTEMA DE VISÃO A LASER PARA MAPEAMENTO DE SUPERFÍCIE DE NAVEGAÇÃO DE ROBÔS QUADRÚPEDES

Tese submetida ao programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisição parcial para a obtenção do Grau de Doutor em Engenharia Elétrica - Automação.

Aprovada em 22 de agosto de 2008.

COMISSÃO EXAMINADORA

**Prof. Dr. Paulo Faria Santos Amaral**
**Universidade Federal do Espírito Santo**
**Orientador**

**Prof. Dr. Antônio Bento Filho**
**Universidade Federal do Espírito Santo**
**Co-orientador**

**Prof. Dr. Mario Sarcinelli Filho**
**Universidade Federal do Espírito Santo**

**Prof. Dr. Edson de Paula Ferreira**
**Universidade Federal do Espírito Santo**

**Prof. Dr. Evandro Ottoni Teatini Salles**
**Universidade Federal do Espírito Santo**

**Prof. Dr. –Ing. Marcelo Ricardo Stemmer**
**Universidade Federal de Santa Catarina**

# Agradecimentos

Agradeço a Deus por minha existência, por tudo de bom que Ele tem propiciado a mim.

À minha esposa Rosilene pelo incentivo total, pelo apoio e pela compreensão dos momentos ausentes devido a este trabalho. Aos meus filhos Ludmila e Stephan, que foram privados de horas de lazer por causa do meu envolvimento com o doutorado.

Ao meu orientador Paulo Faria Santos Amaral, agradeço pela orientação segura, pela habilidade, pela competência e pela grande experiência que enriqueceram este projeto.

Ao meu co-orientador Antônio Bento Filho, pela dedicação e pelo tempo disponibilizado para a orientação, não medindo esforços para me ajudar, com dicas e sugestões preciosas que valorizaram este trabalho.

Aos diretores da Automatica Tecnologia S.A. Benedito Geraldo Miglio Pinto e Rafael Lacerda Alves e seus funcionários, pelo apoio total, proporcionando todos os recursos materiais e humanos no desenvolvimento do Sistema de Visão e do Robô Guará.

Aos professores do Cefetes, Luis Eduardo Martins de Lima, Bene Régis Figueiredo e Reginaldo Corteletti, companheiros que estiveram sempre disponíveis a me ajudar na realização deste projeto.

Ao coordenador do Programa de Pós-Graduação em Engenharia Elétrica da Ufes Mário Sarcinelli Filho, a todos os professores, funcionários e alunos do Del-Ufes, agradeço pela colaboração direta e indireta na realização deste projeto.

# Sumário

## Sumário

# Lista de Figuras

# Lista de Tabelas

# Resumo

Este trabalho apresenta, implementa e testa um sistema de visão a laser, para construção de mapas de superfície de navegação e localização de obstáculos para robôs quadrúpedes. À medida que o robô navega em uma trajetória pré-definida, o sistema mapeia a superfície, identifica os obstáculos e toma decisões para se desviar ou transpô-los, dependendo de suas dimensões, definindo assim as ações de navegação do robô até o seu ponto de destino da trajetória.

O sistema de visão usa uma câmera *CCD* e um sistema laser microcontrolado, que gera linhas de laser para *varrer* a superfície à frente do robô. As imagens adquiridas são então processadas por alguns algoritmos, de modo que os perfis do laser incidindo sobre os obstáculos são usados para gerar imagens de profundidade, que representam mapas 2½D da superfície.

O sistema desenvolvido integra sucessivas imagens de profundidade para extrair dados relevantes dos obstáculos, e, juntamente com os dados de odometria e cinemática do robô, faz o mapeamento da superfície.

Uma estratégia de controle baseada na navegação reativa, que usa uma árvore de decisão binária, foi implementada, para permitir ao robô navegar em superfícies com obstáculos de forma segura e eficiente.

Os resultados experimentais mostraram que o sistema é eficiente. O robô Guará, que foi utilizado nos testes, navegou em ambiente com obstáculos desconhecidos, desviando-se ou transpondo-os, até atingir o ponto de destino da trajetória.

# Abstract

This work presents, implements and tests a laser vision system to build navigating surface maps and localization of obstacles for quadruped robots. As long as the robot navigates on a pre defined trajectory, the system maps the surface, identifies the obstacles and makes decisions to pass by or to pass over them, depending on their dimensions, thus defining the robot navigation actions towards its trajectory destination point.

The vision system uses a *CCD* camera and a laser microcontrolled system, which generates laser stripes *to sweep* the surface in front of the robot. The acquired images are then processed by some algorithms, so that the laser profiles inciding over the obstacles are used to generate range images which represent 2½D surface maps.

The developed system integrates successive range images to extract relevant data from the obstacles and, together with the robot odometry and kinematics data, performs the surface mapping.

A control strategy based on reactive navigation, which uses a binary decision-tree, has been implemented to allow the robot to safely and efficiently navigate on surfaces with obstacles.

Experimental results have shown that the system is efficient. The Guará robot, which has been used in the tests, navigated on an environment with unknown obstacles, passing by or passing over them, up to reaching its trajectory destination point.

# Capítulo 1 – Introdução

## 1.1 Introdução

O Sistema de Visão a Laser para Robôs Quadrúpedes foi desenvolvido para funcionar integralmente com o robô Guará. Este robô é quadrúpede e tem 4 graus de liberdade para cada perna, permitindo-o andar em trajetórias retilíneas e curvilíneas [2]. Ele foi desenvolvido numa parceria entre o Programa de Pós-Graduação em Engenharia Elétrica da Ufes e a empresa Automatica Tecnologia S.A.

Inicialmente concebido sem um sistema de visão, o robô Guará navega por dados de odometria e detecta obstáculos ao seu redor através de micro-chaves localizadas nas suas patas, que são acionadas por contato. Contudo, o robô não tem noção da posição, forma e tamanho destes obstáculos.

O trabalho proposto tem como objetivo dotar o robô Guará de um sistema de visão, usando uma câmera *CCD* e laser para aquisição de imagens à frente do robô, de modo a permitir a identificação de obstáculos através da geração de imagens de profundidade.

Vários algoritmos de aquisição e tratamento de imagens foram implementados para obter as imagens de profundidades precisas e corrigidas, além de sistemas de calibração automática para a câmera CCD e para o laser.

As imagens de profundidade são verdadeiros mapas de superfície, onde ao valor de intensidade $f(x, y)$ de um ponto $(x, y)$ da imagem é atribuída uma profundidade, que pode ser quantificada como um valor de cinza, permitindo o seu armazenamento no formato de imagem.

Através do processo denominado *registro de imagens*, as imagens de profundidade de quadros sucessivos são integradas, formando mapas 2½D da superfície. Devido à limitação

do campo de visão do sistema e da incidência do laser, os obstáculos são gradualmente mapeados e identificados. O sistema proposto extrai destas imagens informações relevantes, conhecidas como descritores, reduz a quantidade de dados, e assim viabiliza computacionalmente o controle do robô.

O sistema proposto processa os dados das imagens integradas, mapeia a superfície de navegação do robô, identifica os obstáculos e, juntamente com os dados de odometria do robô Guará, cria uma estratégia de controle baseada em navegação reativa usando árvore de decisão binária, para permitir que o robô navegue no ambiente de forma segura e otimizada.

O sistema de visão também permite retificar os dados imprecisos de odometria do robô, gerados devido às incertezas nas medições, através de informação visual obtida entre duas imagens sucessivas, usando um algoritmo iterativo para alinhamento de imagens.

Os testes experimentais em superfície com obstáculos mostram que o sistema proposto apresenta ótimo desempenho em funcionamento com o robô Guará.

## 1.2 Definição do Problema

O robô quadrúpede Guará não possui um sistema de visão. Trata-se de um robô com uma plataforma e 4 pernas, com 4 graus de liberdade em cada uma, que lhe permite andar em trajetória retilínea e curvilínea.

O robô Guará foi criado para realização de tarefas específicas pertinentes aos robôs quadrúpedes, como andar em ambientes com superfície irregular, subir e descer degraus. Ele usa uma andadura baseada em equilíbrio estático de tal forma que 3 patas são alternadamente mantidas no solo e apenas uma se encontra em vôo.

No sistema atual, uma vez que o robô esteja andando em uma trajetória pré-definida, ele só detecta obstáculos ao seu redor quando os toca. Micro-chaves localizadas nas patas são acionadas por contato. Não é possível para o robô discernir se ele pode transpor tais obstáculos, porque ele não tem informação alguma da superfície de navegação.

Na presença de obstáculos baixos ou altos, ele só os detecta quando os toca com as patas. No atual estágio, ele recolhe a pata que toca o obstáculo, eleva-a para uma altura máxima e tentar transpô-lo na hora do passo. Caso o obstáculo seja largo demais, o passo será insuficiente e o robô descerá a pata sobre o obstáculo, não realizando a tarefa com sucesso. Atualmente o robô Guará dispõe de algoritmos com realimentação tátil para superar alguns tipos de obstáculos com dimensões compatíveis com os seus limites cinemáticos.

O robô Guará não conhece detalhes intrínsecos da superfície, como a posição, forma e tamanho dos obstáculos. Portanto, durante a sua navegação, os obstáculos podem ser tocados, e assim causar inconvenientes à navegabilidade do robô. Em certos casos, isto poderia até inviabilizar o seu uso.

## 1.3 Solução Proposta

O robô Guará deve ser dotado de *um sistema de visão* para mapeamento da sua superfície de navegação e identificação dos obstáculos, de modo que o sistema de controle do robô atue sobre as patas, modificando as suas trajetórias sempre que houver obstáculos que o impeçam de navegar na trajetória pré-definida.

O sistema proposto deve ser capaz de adquirir imagens de profundidade, para tornar possível o mapeamento da superfície. Através destas imagens, devem ser extraídos descritores que as representem por um conjunto reduzido de dados de viabilidade computacional. Descritores, como largura, comprimento, altura, área, volume e centróide dos obstáculos, são dados relevantes que ajudam na tarefa de decidir pela transposição ou não de obstáculos pelas pernas do robô.

O sistema proposto deve trocar mensagens com o sistema de controle do robô para obtenção de dados cinemáticos e de odometria. Assim, o sistema deve tomar decisões de modificação de trajetória, gerando ações corretivas e repassando-as para o sistema de controle do robô. O sistema proposto deve assim garantir que o robô possa navegar de forma segura e eficiente em ambientes semi-estruturados, com obstáculos desconhecidos.

## 1.4 Objetivos

O objetivo do trabalho é o desenvolvimento de um sistema de visão, para aquisição de imagens de profundidade da superfície, extração de dados descritores, identificação de obstáculos, comunicação com sistema de controle do robô para obtenção de dados de odometria e criação de uma estratégia de controle para tomadas de decisão de modo que o robô possa navegar em superfícies com obstáculos de forma segura e eficiente.

Para a viabilização deste sistema, os seguintes tópicos foram abordados:

- ✓ estudo de robôs móveis, em especial robôs quadrúpedes;

- ✓ estudo de sistemas sensoriais e de visão para robôs móveis;
- ✓ estudo de algoritmos de aquisição e tratamento de imagens;
- ✓ projeto de um sistema de visão com melhor desempenho para o robô quadrúpede Guará;
- ✓ implementação de um sistema computacional usando programação orientada a objetos em linguagem de programação de alto nível;
- ✓ desenvolvimento de algoritmos de aquisição e tratamento de imagens;
- ✓ desenvolvimento de algoritmos para geração de imagens de profundidade;
- ✓ desenvolvimento de algoritmos para extração de descritores de obstáculos;
- ✓ desenvolvimento de um método para mapeamento da superfície;
- ✓ desenvolvimento de uma estratégia de controle para tomadas de decisão;
- ✓ implementação e testes do sistema proposto;
- ✓ integração do sistema proposto com o sistema de controle do robô;
- ✓ testes de navegação do robô com o sistema proposto integrado;
- ✓ análise de desempenho do sistema proposto.

## 1.5 Contribuições desta Tese

Esta tese descreve a *Concepção e a Implementação de um Sistema de Visão a Laser e uma Metodologia para Mapeamento da Superfície de Navegação de Robôs Quadrúpedes*, um trabalho de pesquisa e desenvolvimento em robótica móvel na área de visão, que traz as seguintes contribuições:

*Principal:*

*"Uma metodologia para aquisição, tratamento e condicionamento de imagens, geração de imagens de profundidade e registro, criação e extração de dados descritores dos obstáculos e mapeamento da superfície de navegação de robôs quadrúpedes em ambientes semi-estruturados".*

*Secundárias:*

- ✓ o *desenvolvimento e construção de um sistema de visão microcontrolado*, usando laser e uma câmera *CCD,* baseado em triangulação ativa;

- ✓ um *ambiente de software integrado*, usando programação orientada a objetos (POO), para aquisição, tratamento e condicionamento de imagens; geração, armazenamento e registro de imagens de profundidade; extração de descritores dos obstáculos; mapeamento de superfície de navegação e implementação de uma técnica de controle de navegação reativa para o robô Guará;

- ✓ um *sistema de calibração automática para câmeras CCD*, para cálculo dos seus parâmetros intrínsecos, com correção da distorção radial, interpolação bilinear e usando geometria projetiva;

- ✓ um *sistema de calibração automática para geração de imagens de profundidade*, baseado em triangulação ativa com laser, usando um processo de orto-retificação para correção do efeito de perspectiva das imagens;

- ✓ uma técnica adaptada do *algoritmo iterativo do ponto mais próximo (ICP – Iterative Closest Point)*, para refinamento do registro de imagens de profundidade sucessivas e correção dos dados de odometria do robô quadrúpede;

- ✓ um *algoritmo* para redução de pontos *3D* de imagens de profundidade, denominada *cubificação*, segmentação dos obstáculos e geração dos seus descritores;

- ✓ um *sistema de controle baseado em navegação reativa*, usando árvore de decisão binária, para a navegação do robô quadrúpede em ambientes com obstáculos desconhecidos.

## 1.6 Estrutura da Tese

Esta Tese está dividida em 7 capítulos, como segue:

- **Capítulo 1 – Introdução**

Neste capítulo introdutório, são mostrados a definição do problema a ser tratado, a solução proposta, os objetivos do sistema proposto, a contribuição desta tese e a sua estruturação.

- **Capítulo 2 – Técnicas de Medição de Distância Usadas em Navegação de Robôs**

O capítulo 2 apresenta as principais técnicas de medição de distância usadas para navegação de robôs móveis, discriminando as suas características, aplicações e limitações. Após este estudo minucioso é definida a técnica de medição de distância, a ser adotada pelo sistema proposto, que melhor se adapta para o uso com o robô Guará.

- **Capítulo 3 – Arquitetura e Modelagem do Sistema de Visão Proposto**

Este capítulo aborda a arquitetura e modelagem do sistema proposto. Este sistema é dividido em 4 módulos distintos, e cada qual é detalhado, mostrando suas conexões com os outros módulos e suas principais funcionalidades. Além do mais, neste capítulo são mostrados os algoritmos de aquisição e tratamento de imagens, correção de distorção radial, geometria projetiva e projeção ortogonal, e geração de imagens de profundidade.

- **Capítulo 4 – Mapeamento da Superfície de Navegação do Robô Guará**

Neste capítulo são descritos os sistemas de coordenadas do robô Guará e do sistema de visão proposto, e as transformações entre estes sistemas. Estas transformações são usadas para registro (alinhamento) de imagens de profundidade. O algoritmo iterativo do ponto mais próximo *ICP* é usado para refinamento do registro de imagens e correção dos dados da trajetória do robô. A modelagem de andadura do robô Guará é descrita e um algoritmo

de cubificação é criado para auxiliar na geração de vários descritores dos obstáculos da superfície de navegação do robô Guará.

- **Capítulo 5 – Sistema de Controle de Navegação do Robô Guará**

Neste capítulo é detalhada a estratégia de controle proposta para as tomadas de decisão do robô durante a navegação. O método proposto usa *Navegação Reativa,* com um sistema baseado em Árvore de Decisão Binária, com regras de percepção e ação que permite ao robô transpassar ou desviar-se de obstáculos. A arquitetura de controle e sua modelagem são implementadas. Neste capítulo a comunicação e a integração do sistema proposto com o sistema de controle do robô Guará são tratadas.

- **Capítulo 6 – Resultados Experimentais**

Neste capítulo são feitas análises de desempenho do sistema para obtenção de resultados experimentais. Testes de geração de imagens de profundidade, de mapeamento da superfície e de controle de navegação do robô, com situações de obstáculos na superfície, são executados.

- **Capítulo 7 – Conclusões**

De posse de resultados experimentais do sistema, são apresentadas as conclusões desta tese, além de sugestões de melhorias futuras nesta linha de pesquisa.

# Capítulo 2 – Técnicas de Medição de Distância Usadas em Navegação de Robôs

## 2.1 Introdução

Escolher uma boa técnica de medição de distância é definir um bom sistema sensorial que faça com que os robôs tenham potencial para fazer tarefas cada vez mais sofisticadas, podendo operar em ambientes internos e externos, estruturados e não-estruturados, adaptando-se às mudanças ao seu redor.

A habilidade que um robô tem em identificar o mundo ao seu redor e mudar o seu comportamento o torna interessante e poderoso na execução destas tarefas. Conseqüentemente, à medida que a tecnologia avança, os robôs exigem novas técnicas que acompanhem as suas necessidades operacionais.

As técnicas de medição de distância para robôs móveis variam desde uma simples medição pontual até a geração de mapas *3D* do ambiente de navegação do robô. Sistemas sensoriais cada vez mais rápidos, eficientes e precisos devem ser capazes de extrair informações do ambiente que auxiliem os robôs na definição da orientação, posição, velocidade, aceleração, detecção de obstáculos, etc.

Há técnicas de medição de distância que usam câmeras, permitindo medições de toda uma superfície, como, por exemplo, a *visão estéreo*. Porém, estas técnicas necessitam de grande esforço computacional, pois o tempo gasto para aquisição, armazenamento e execução de algoritmos de tratamento de imagens é significativo, limitando assim o seu uso em aplicações de tempo real.

Em muitos sistemas baseados em câmeras, uma reconstrução *3D* não é necessária, sendo substituída por uma série de dados do ambiente. Por exemplo, a informação se há um obstáculo à frente do robô, não importando a forma, cuja altura seja transponível pelos

limites cinemáticos do robô. Os dados dos obstáculos são descritores, que, por serem um conjunto reduzido, têm custo computacional menor, viabilizando aplicações em tempo real.

Este capítulo descreve as principais técnicas de medição de distância empregadas em sistemas sensoriais para robótica, com o foco direcionado para sistemas que medem distâncias reais entre um referencial e um alvo. O objetivo é definir, entre várias técnicas, qual a ideal para uso na plataforma do robô Guará. A técnica escolhida deve permitir que o robô identifique e reconheça os obstáculos ao seu redor, mapeando-os de modo que ele possa navegar em ambientes semi-estruturados de forma segura e eficiente.

## 2.2 Técnicas de Medição de Distâncias

Sistemas que medem a distância entre um referencial e um alvo de interesse, sem contato físico, são muito usados em robótica. Há várias técnicas empregadas em várias implementações. A Figura 2.1 mostra as principais técnicas de medição de distâncias usadas em sistemas sensoriais para robótica [5].

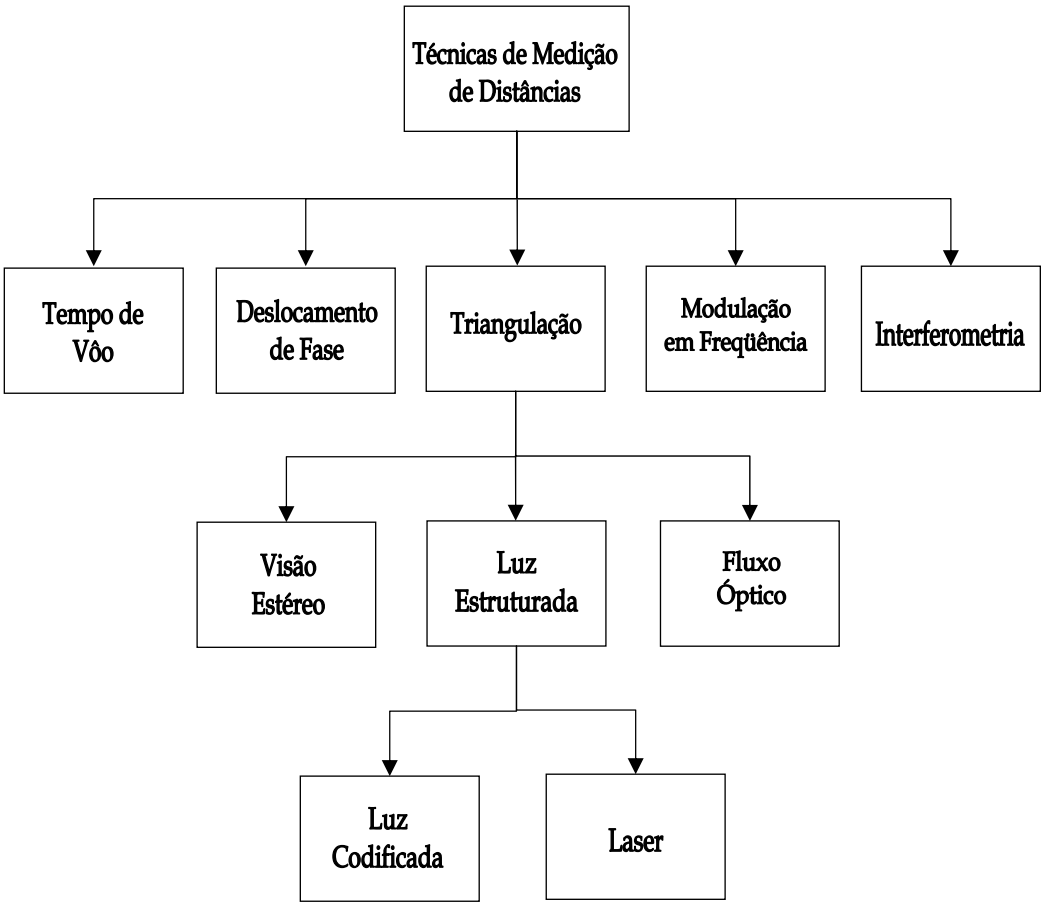


*Figura 2.1: Principais técnicas de medição de distâncias em robótica.*

Os sensores de medição de distância sem contato são considerados *ativos* quando irradiam algum tipo de energia (geralmente luz) sobre a região de interesse para a medição, e são considerados *passivos* quando a energia captada pelos sensores é proveniente dos objetos ou superfície sob medição.

Na Figura 2.1 há sistemas com sensores ativos e passivos. Estes sistemas, com suas principais características e funcionalidades, são discutidos a seguir.

### 2.2.1 Medição de Distância por Tempo de Vôo

Esta técnica de medição baseia-se em medir o tempo decorrido entre o envio de um sinal pulsado, geralmente ondas eletromagnéticas ou sonoras, e a recepção do eco deste sinal causado por um determinado objeto, conforme mostrado na Figura 2.2.

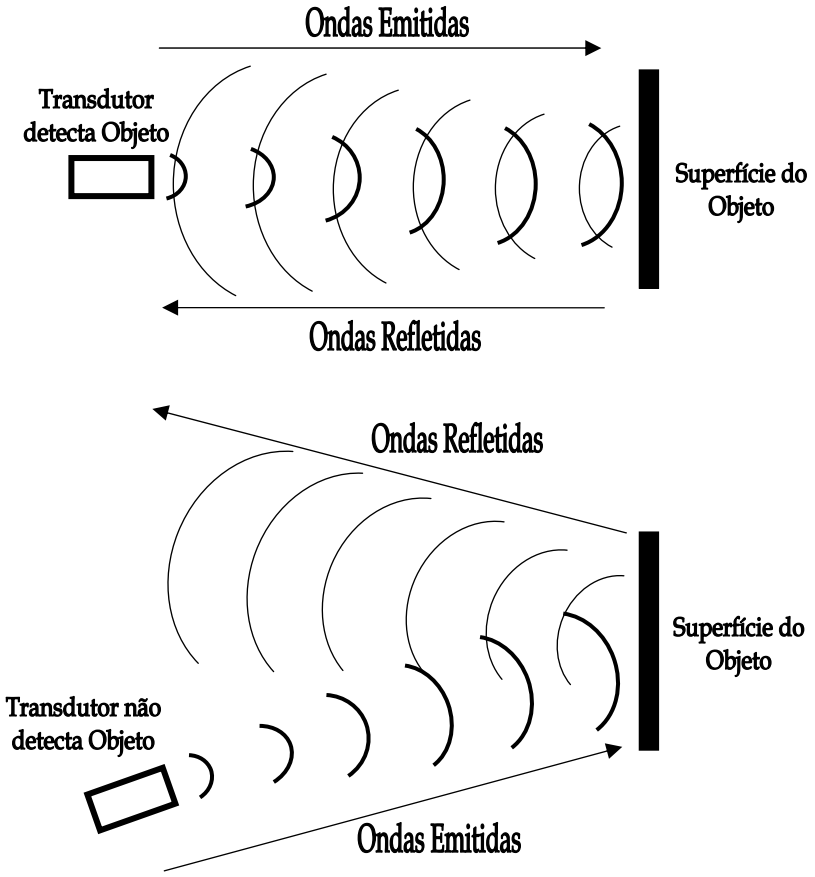


*Figura 2.2: Medição de distância por tempo de vôo.*

Seja $t$ o tempo decorrido entre a transmissão inicial e a detecção do eco, $d$ a distância entre o referencial e o objeto e $v$ a velocidade da onda. A distância $d$ é dada por

$$d = \frac{v.t}{2}. \tag{2.1}$$

A velocidade do sinal é conhecida e constante dentro de um determinado meio físico. Esta técnica de medição ativa é comumente usada em *radar* (*radio direction and ranging*), *sonar* (*sound navigation and ranging*) e *ladar* (*laser direction and ranging*).

Os sensores sonares têm sido largamente usados em robôs móveis autônomos por muitos anos. A velocidade do som é baixa o suficiente para ser facilmente medida com equipamentos de baixo custo. Estes sistemas alinham boa disponibilidade e facilidade de conexão a computadores e atualmente são usados para que robôs evitem colisão com obstáculos.

Os sensores sonares são também usados para construção de mapas, localização e navegação de robôs. O mais popular sensor sonar usado é o *rangefinder* ultra-sônico *Polaroid*, que mede o tempo de vôo com um único transdutor que atua como transmissor e receptor [7].

É conhecido que sensores sonares sofrem alguns problemas severos: *baixa resolução angular*, *resolução de faixa limitada*, *reflexões especulares* e erros de leitura freqüentes devido às fontes externas de ultra-som e *crosstalk* [8].

Os sensores ultra-sônicos têm algumas vantagens sobre o *laser* e câmera, devido ao baixo consumo, baixo custo e independência de luz. Além do mais, o seu sinal tem baixa largura de banda e, portanto, os requerimentos para processamento do sinal são reduzidos. Por outro lado, a temperatura ambiental afeta a absorção da onda; a grande largura do feixe (de 30 a 60º) do transdutor ultra-sônico conduz a medições angulares imprecisas; a baixa largura de banda também significa baixa resolução da distância, fazendo com que objetos próximos fiquem indistinguíveis [9].

### 2.2.2 Medição de Distância por Deslocamento de Fase

A técnica de medição por deslocamento de fase envolve uma transmissão contínua da onda, diferentemente da técnica de tempo de vôo, que envia um sinal pulsado de curta duração.

Uma vantagem desta técnica é a possibilidade de medir, além da distância, a direção e velocidade de um alvo em movimento [5]. Também utilizam esta técnica o *radar* e o *ladar*.

A técnica se baseia no efeito Doppler: "*a freqüência de uma onda de energia refletida de um objeto em movimento é uma função da velocidade relativa entre o objeto e o observador*". A Figura 2.3 ilustra o funcionamento desta técnica.

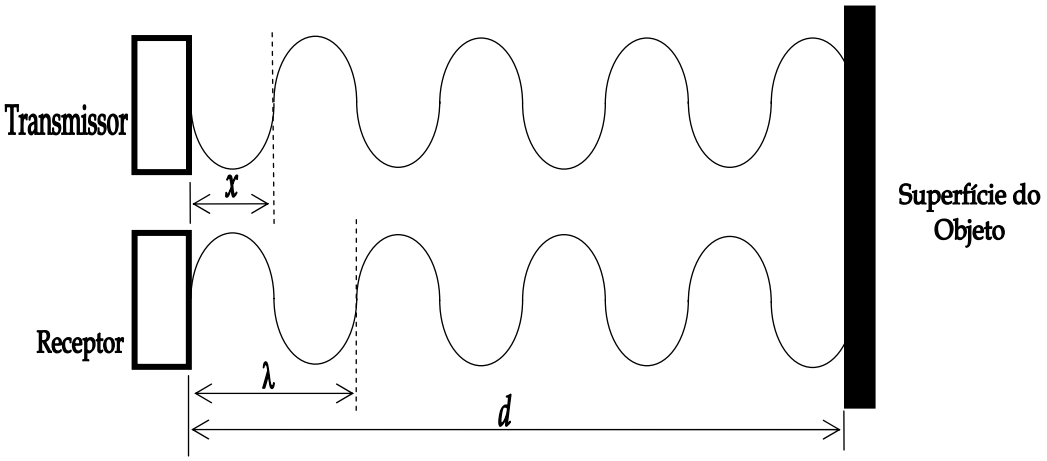


*Figura 2.3: Medição de distância por deslocamento de fase.*

O deslocamento de fase, expresso como uma função da distância em relação à superfície do objeto, é dado por

$$\phi = \frac{4\pi.d}{\lambda}, \tag{2.2}$$

onde $\phi$ é o deslocamento de fase, $d$ é a distância até o objeto e $\lambda$ é o comprimento de onda do sinal modulado. Na Figura 2.3, $x$ é a distância correspondente ao ângulo de defasagem $\phi$ entre o sinal modulado enviado e o sinal refletido pelo objeto.

Portanto, a distância $d$ pode ser expressa como

$$d = \frac{\phi.\lambda}{4\pi} = \frac{\phi.c}{4\pi.f}, \tag{2.3}$$

onde $c$ é a velocidade da luz e $f$ é a freqüência do sinal modulado.

### 2.2.3 Medição de Distância por Modulação de Freqüência

Esta técnica envolve a transmissão de uma onda eletromagnética contínua modulada por um sinal triangular periódico que ajusta a freqüência portadora (*carrier*) acima ou abaixo da freqüência média $f_o$ [5]. O transmissor emite um sinal que varia em freqüência como uma função linear do tempo,

$$f(t) = f_o + a.t, \tag{2.4}$$

onde *a* é uma constante e *t* é tempo decorrido.

O sinal refletido de um objeto chega ao receptor a um tempo $t + T$ , onde,

$$T = \frac{2d}{c}, \tag{2.5}$$

sendo que $T$ é tempo de propagação de ida e volta da onda, $d$ é a distância até o objeto e $c$ é a velocidade da luz. A Figura 2.4 mostra a onda modulada por um sinal triangular periódico.

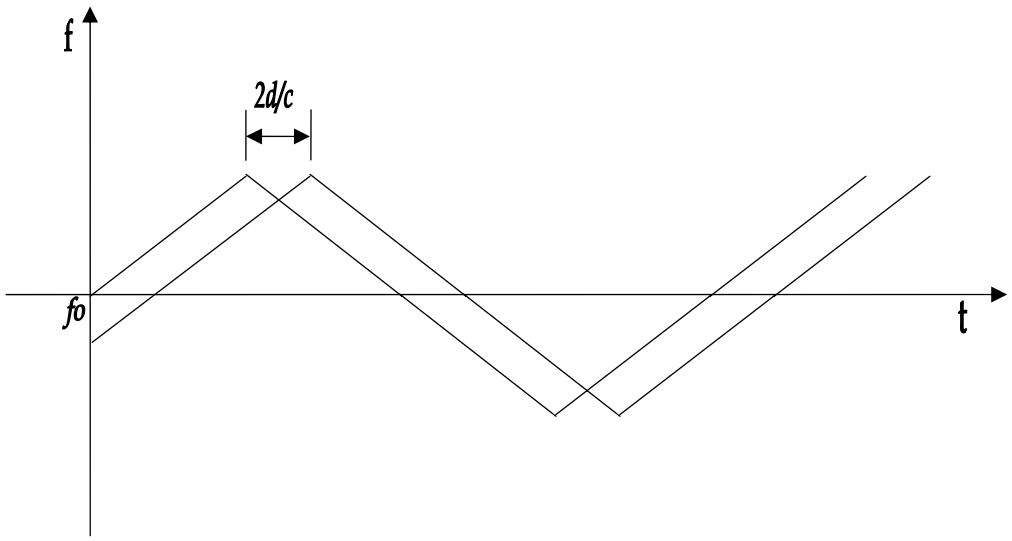


*Figura 2.4: Medição de distância por modulação de freqüência.*

As duas freqüências, quando combinadas, produzem uma freqüência de batimento $F_b$, dada por

$$F_b = f(t) - f(T+t) = a.T. \tag{2.6}$$

A freqüência de batimento é medida e usada para calcular a distância até o objeto, dada por

$$d = \frac{F_b.c}{4F_r.F_d}, \tag{2.7}$$

onde $F_r$ é a freqüência de modulação e $F_d$ é o desvio total da freqüência modulada.

Esta técnica tem a vantagem de não apresentar medição de distância ambígua, o que pode ocorrer no caso da técnica de deslocamento de fase. Contudo ela apresenta algumas desvantagens associadas com a linearidade e repetibilidade da rampa de freqüência, e problema de coerência do feixe de laser dos sistemas óticos. Como conseqüência, a maior parte dos sistemas de medição com esta técnica é usada em radares, enquanto que sistemas com laser usam comumente a técnica de tempo de vôo [5].

### 2.2.4 Medição de Distância por Interferometria

A interferometria é um dos métodos de medição de distância mais precisos. Inicialmente esta técnica era usada somente em laboratórios, devido ao controle das condições ambientais nestes locais, mas graças aos recentes desenvolvimentos da tecnologia ótica instrumentos baseados na interferometria podem ter uso externo.

O princípio de funcionamento desta técnica baseia-se em padrões de interferência resultantes quando duas ondas que viajam em caminhos diferentes são comparadas. A Figura 2.5 mostra um diagrama de funcionamento desta técnica.

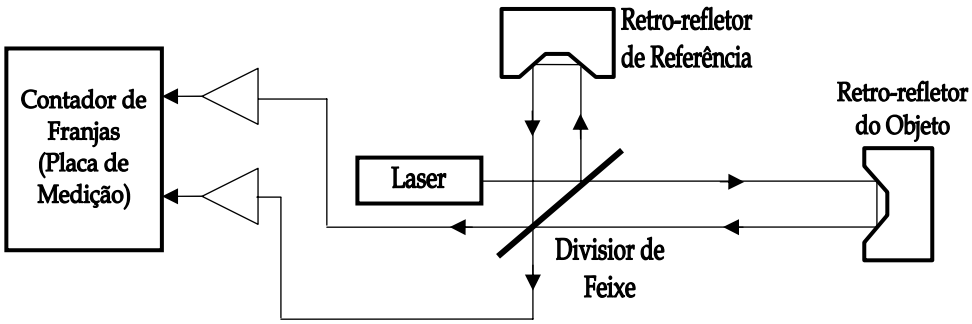


*Figura 2.5: Medição de distância por interferometria.*

O feixe de laser passa por um divisor de feixes, dividindo-se em *feixe de referência* e *feixe de medição*. O feixe de referência é direcionado para o retro-refletor onde é refletido e vai para a placa de medição. O feixe de medição viaja através do ar até o retro-refletor preso ao objeto de interesse (é necessário usar retro-refletores nos objetos para garantir um sinal de retorno confiável [5]).

Como os caminhos percorridos pelos dois feixes são diferentes, eles se combinarão de tal forma que interferências construtivas e destrutivas são produzidas (*franjas*). As *franjas* são

padrões de distúrbios na forma de onda combinada que altera entre intensidade máxima e mínima.

Ao contar a quantidade de *franjas* que passam pelo detector, é possível calcular com extrema precisão a distância até o retro-refletor (do objeto) ao longo da linha do feixe de laser.

Os interferômetros não medem a distância absoluta, mas a distância relativa que um objeto está se movimentando em relação a um referencial. Um deslocamento do objeto em 1 centímetro resulta no movimento de aproximadamente 10 milhões de *franjas*.

Embora extremamente preciso, o sistema é afetado por diferentes fatores. São eles: *erros ambientais* (o índice de refração do feixe de laser é afetado por variações na temperatura ambiente, pressão do ar e umidade), *erros geométricos* (desalinhamentos do sistema ótico) e *erros de Instrumentos* [10].

### 2.2.5 Medição de Distância por Triangulação

A técnica de medição de distância por triangulação é amplamente usada em vários instrumentos e sistemas de medição de distância. Diversas variantes desta técnica foram implementadas e as principais são abordadas neste item. A triangulação baseia-se numa importante premissa da trigonometria plana, que declara que dado o comprimento de um lado e dois ângulos de um determinado triângulo, é possível determinar o comprimento dos outros dois lados deste triângulo. A Figura 2.6 mostra o princípio da triangulação.

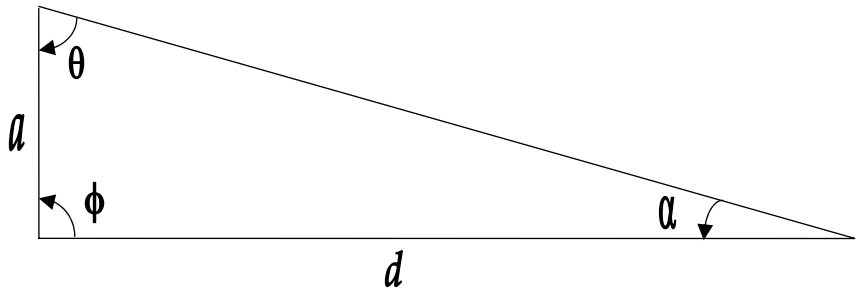


*Figura 2.6: Medição de distância por triangulação.*

A Lei dos Senos pode ser rearranjada para definir *d* em relação a *a*, como

$$d = \frac{a.sen\theta}{sen\alpha} = \frac{a.sen\theta}{sen(\theta + \phi)}. \tag{2.8}$$

Em sistemas de medição com esta técnica, a distância *a* é conhecida como linha base de separação do sensor. O lado *d* representa a distância a ser medida.

Os sistemas de medição por triangulação podem ser classificados em *passivos* (usam a luz ambiente) ou *ativos* (usam luz própria para iluminar os objetos). Ambos sistemas têm os seus prós e contras.

A triangulação passiva usa geralmente câmeras de vídeo que requerem condições de iluminação ambiente adequadas, se o ambiente for escuro, enquanto que a triangulação ativa não requer iluminação ambiente especial, mas por outro lado pode sofrer com problemas de reflexão e absorção do sinal pela superfície.

Fatores de limitação comuns a ambas triangulações incluem redução da precisão à medida que a distância medida aumenta, erros de medição angular e partes perdidas devido ao problema de *sombra* dos objetos.

A seguir são abordados os principais métodos de medição de distância por triangulação.

### 2.2.5.1 Visão Estéreo

Esta técnica de triangulação passiva modela o comportamento biológico do sistema de visão dos seres humanos. Quando um objeto tridimensional é visto de dois pontos de um plano normal à direção da visão, a imagem observada por um ponto é deslocada lateralmente em relação à imagem observada pelo outro ponto. Este deslocamento é chamado de *disparidade* e é tanto menor quanto mais distante estiver o objeto.

A partir de duas câmaras posicionadas de forma adequada, com suas posições e direcionamentos conhecidos, é possível determinar a posição de qualquer ponto neste espaço, desde que este ponto possa ser localizado dentro de cada uma das imagens capturadas pelas câmaras. Por isso, é necessário haver uma região comum, de proporção razoável, aparecendo em ambas as imagens. A determinação da posição de um ponto dentro deste espaço é obtida por triangulação. Define-se *stereopsis* como a capacidade de determinação da distância de profundidade baseada na informação de um par estéreo.

Existem 4 passos básicos envolvendo este processo [5], a saber,

- ✓ identificação de um ponto de interesse em uma imagem;
- ✓ o mesmo ponto de interesse deve ser localizado na outra imagem;
- ✓ as posições laterais de ambos os pontos devem ser medidas em relação a uma referência comum;
- ✓ a distância é, então, calculada pela disparidade nas medidas laterais.

A identificação de pontos correspondentes nas duas imagens é complicada em regiões onde a intensidade e cor são uniformes. Outro agravante é a presença de região de sombra em uma imagem em relação à outra. A variação da luz ambiente, com comportamento diferente nas duas imagens, dificulta encontrar o ponto de correspondência entre elas.

A Figura 2.7 mostra a geometria epipolar usada neste modelo. As câmeras são modeladas segundo o modelo pontual de câmera (*pinhole model*).

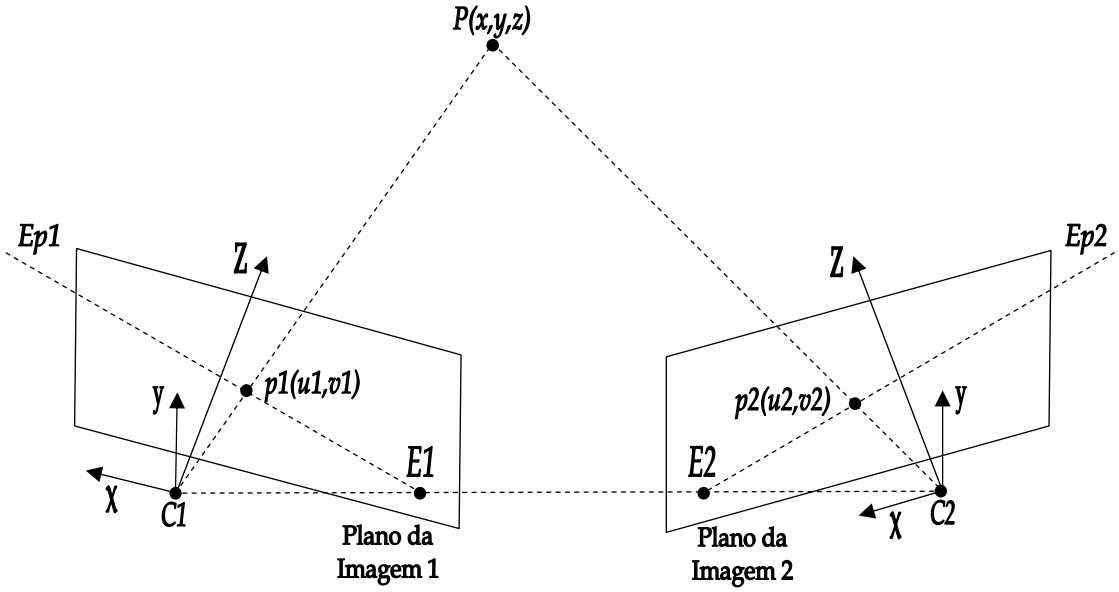


*Figura 2.7: Geometria epipolar.*

A geometria epipolar é uma técnica que reduz o número da dimensão de procura de dois para 1, cujo objetivo é minimizar o esforço computacional para encontrar os pontos de correspondência entre as imagens.

Um ponto *3D* $P = (x, y, z)$ é projetado em cada imagem, originando os pontos $p_1 = (u_1, v_1)$ e $p_2 = (u_2, v_2)$. O plano epipolar é definido pelo ponto $P$ e os dois centros óticos $C_1$ e $C_2$ das câmeras. Este plano $PC_1C_2$ intercepta os dois planos das imagens em duas retas: $Ep_1$ e $Ep_2$, denominadas de retas epipolares. A reta $Ep_1$ passa por $E_1$ e $p_1$ e a reta $Ep_2$ passa por $E_2$ e $p_2$. $E_1$ e $E_2$ são denominados de pontos epipolares e são os pontos de interseção da linha base $C_1C_2$ com cada plano de imagem. Desta maneira, para um determinado ponto de interesse na imagem 1 pertencente à reta epipolar $Ep_1$, o seu ponto correspondente na imagem 2 deve estar sobre a reta epipolar $Ep_2$, e assim o espaço de

procura pela correspondência é reduzido de dimensão 2 (plano) para 1 (reta), tornando a procura unidimensional.

Sejam $\tilde{\mathbf{p}}_1 = \begin{bmatrix} u_1 & v_1 & 1 \end{bmatrix}$ e $\tilde{\mathbf{p}}_2 = \begin{bmatrix} u_2 & v_2 & 1 \end{bmatrix}$ vetores homogêneos representando os pontos $p_1$ e $p_2$. O principal resultado da geometria epipolar é que a seguinte relação linear pode ser escrita:

$$\tilde{\mathbf{p}}_1^T . \mathbf{F} . \tilde{\mathbf{p}}_2 = 0. \tag{2.9}$$

Portanto $F$ é a conhecida *matriz fundamental*, que é uma entidade 3x3, com 9 parâmetros que representam os parâmetros intrínsecos de ambas as câmeras.

Existem vários métodos para estimar a *matriz fundamental* [11], [12] e também o uso da geometria epipolar diferencial [13].

Para aquisição de dados de distância (*range*), [14] aborda o uso da visão estéreo com iluminação estruturada, enquanto [15] usa a visão estéreo para mapeamento e detecção de características do ambiente.

Há muitos artigos e livros abordando a visão estéreo, cujo foco maior é na área de inteligência artificial, sistemas autônomos com robôs e processamento de imagens para sistemas de visão.

#### 2.2.5.2 Fluxo Óptico

Técnica de triangulação passiva que usa apenas uma câmera. O fluxo óptico é a distribuição *2D* das velocidades aparentes do movimento do padrão de intensidade no plano da imagem, devido ao movimento relativo entre a câmera e a cena [16]. Em outras palavras, o campo do fluxo óptico consiste de um campo denso de velocidade onde a cada *pixel* no plano da

imagem está associado um único vetor de velocidade. Esta técnica é usada quando o robô está se movimentando em relação ao ambiente de análise.

Para fins de visualização, o campo do fluxo ótico é mostrado em um mapa chamado de mapa de agulhas (*needle map*). Em se conhecendo o intervalo de tempo entre duas imagens consecutivas, os vetores da velocidade podem ser convertidos em vetores de deslocamento e vice-versa. A Figura 2.8 mostra um mapa de agulhas de uma cena.

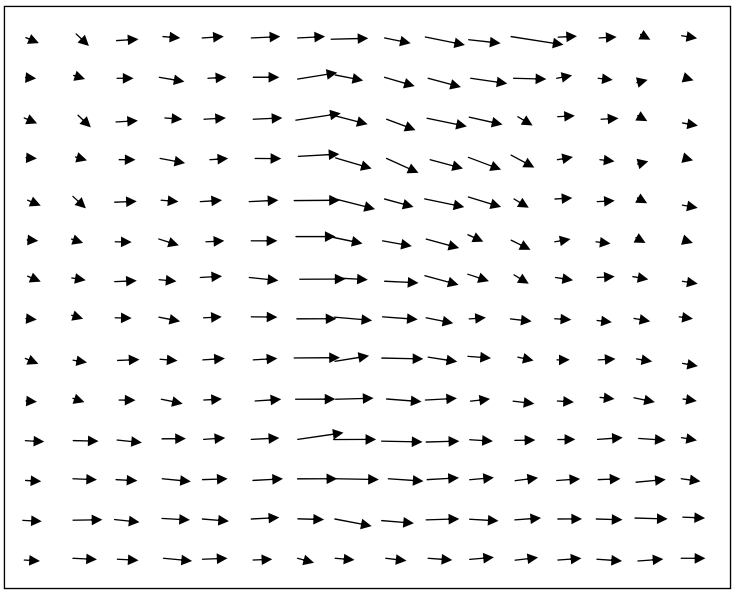

*Figura 2.8: Mapa de agulhas representando o fluxo óptico.*

O fluxo calculado fornece uma boa noção sobre a organização espacial dos objetos e permite a distinção entre caminho livre e obstáculos. As regiões com baixa variação no padrão de intensidade podem ser consideradas como passagens livres, enquanto que as regiões com alta mudança dos padrões de intensidade são consideradas possíveis obstáculos.

O cálculo do fluxo óptico ou velocidade da imagem é um problema fundamental no processamento de seqüências de imagens, e pode auxiliar em várias tarefas, tais como interpretação de cena, navegação exploratória, acompanhamento de objetos, avaliação de tempo para colisão, segmentação de objetos, codificação de vídeo, visão de robôs, etc.

Como apresentado em [17], os métodos de computação do fluxo óptico podem ser classificados em 4 grandes grupos, a saber,

- ✓ métodos diferenciais;
- ✓ métodos baseados em freqüência;
- ✓ métodos baseados em correlação;
- ✓ métodos de múltiplos movimentos.

Nos métodos diferenciais, a hipótese inicial para a computação do fluxo óptico é a de que a intensidade entre quadros sucessivos é aproximadamente constante em um intervalo de tempo pequeno, ou seja, neste pequeno intervalo, o deslocamento é mínimo.

A velocidade da imagem é computada a partir das derivadas espaço-temporais da intensidade na imagem. Considerando uma imagem I em tons de cinza, pode-se chegar à expressão [16]

$$\mathbf{I}(\vec{\mathbf{x}},t) \approx \mathbf{I}(\vec{\mathbf{x}}+\delta\vec{\mathbf{x}},t+\delta t), \tag{2.10}$$

onde $\mathbf{I}(\vec{\mathbf{x}},t)$ é a função de intensidade da imagem, $\vec{\mathbf{x}} = \begin{bmatrix} x & y \end{bmatrix}$ é o vetor posição na imagem e $\delta\vec{\mathbf{x}}$ é o deslocamento de uma região da imagem em $(\vec{\mathbf{x}},t)$ após o tempo $\delta t$.

Expandindo o lado direito da equação (2.10) por série de Taylor:

$$\mathbf{I}(\vec{\mathbf{x}}+\delta\vec{\mathbf{x}},t+\delta t) = \mathbf{I}(\vec{\mathbf{x}},t) + \nabla\mathbf{I}\delta\vec{\mathbf{x}} + \delta t\mathbf{I}_{\mathbf{t}} + \mathbf{O}^2, \tag{2.11}$$

onde, $\nabla\mathbf{I} = \begin{bmatrix} I_x & I_y \end{bmatrix}$ é o gradiente nas direções *x* e *y*, $\mathbf{I}_t$ é a derivada parcial de primeira ordem em relação ao tempo de $\mathbf{I}(\vec{\mathbf{x}},t)$ e $\mathbf{O}^2$ são os termos de segunda ordem em diante, que são desprezados.

Subtraindo-se $\mathbf{I}(\vec{\mathbf{x}},t)$ em ambos os lados e dividindo-se por $\delta t$, obtém-se:

$$\nabla \mathbf{I}.\frac{\delta \vec{\mathbf{x}}}{\delta t} + \mathbf{I}_t = 0 \Rightarrow \nabla \mathbf{I}.\vec{\mathbf{v}} + \mathbf{I}_t = 0, \tag{2.12}$$

onde $\vec{\mathbf{v}} = \begin{bmatrix} \frac{\delta x}{\delta t} & \frac{\delta y}{\delta t} \end{bmatrix}$ é o campo de velocidades na imagem. A equação (2.12) é chamada de *equação de restrição do fluxo óptico* e define uma restrição local e única sobre o movimento da imagem [16]. Esta restrição não é suficiente para determinar as componentes de $\vec{v}$, pois a equação (2.12) possui duas incógnitas, tendo infinitas soluções. Este problema é conhecido como *problema de abertura* [18].

A precisão das técnicas diferenciais, depende, principalmente da estimativa das derivadas parciais da função da intensidade. Embora o método de diferenças finitas seja simples, ele não consegue distinguir entre dados verdadeiros e ruído. Para reduzir o efeito negativo do ruído, a imagem passa por um filtro gaussiano previamente.

Há outras metodologias para o cálculo do fluxo óptico na literatura. O método de Horn e Schunck [16] usa uma forma de regularização aplicada à equação (2.12), conhecida como *restrição de suavização*, significando que os vetores de fluxo variam de uma imagem para a outra de forma suave.

#### 2.2.5.3 Luz Estruturada

Sistemas que empregam esta técnica são casos mais refinados de triangulação ativa. O princípio de funcionamento baseia-se em projetar um padrão de luz (seja uma linha, um *grid* de luz ou pontos de luz) sobre a superfície dos objetos, enquanto que uma câmera, em triangulação com o emissor da luz estruturada, captura este padrão modificado pelos objetos.

A informação de distância entre a câmera e o objeto se manifesta através das distorções visíveis no padrão projetado devido às variações nas profundidades da cena observada.

O uso destes efeitos especiais de luz tende a reduzir a complexidade computacional e melhorar a confiabilidade na análise de objetos *3D* [5].

A configuração mais comum de luz estruturada é a projeção de uma linha de luz sobre a cena, cuja fonte de luz mais usual é o laser. Onde a linha intercepta um objeto, a câmera captura os deslocamentos na linha de laser que são proporcionais à profundidade da cena.

A Figura 2.9 mostra um sistema típico que usa esta técnica. Para se obter informações mais densas da imagem, o plano de laser tem que se mover através da cena.

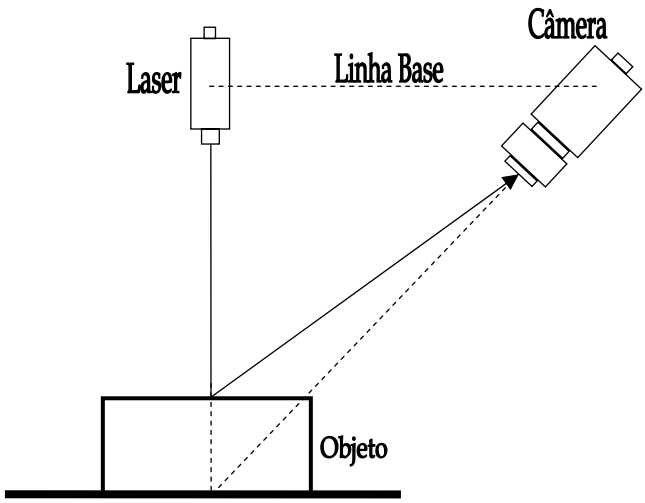


*Figura 2.9: Medição de distância com luz estruturada.*

A relação entre o deslocamento da linha de laser e a distância a ser medida é dependente do comprimento da linha base entre o emissor de laser e a câmera. Como em qualquer sistema de triangulação, à medida que a linha base aumenta, a precisão do sistema aumenta. Contudo o problema de sombras (regiões não visíveis da cena) aumenta.

Para obtenção da informação *3D* de distância de toda uma cena usando a técnica da Figura 2.9, é necessário que o plano de laser *varra* toda a superfície a se mapear. A desvantagem deste método é que a extração das distâncias demanda tempo, e é difícil devido à necessidade de armazenar e analisar vários quadros.

Os sensores de luz estruturada geram pontos densos, com bastante informação da superfície e evitam o problema de correspondência existente na técnica da visão estéreo, tornando este método atrativo para muitas tarefas de inspeção de visão [19]. Hoje em dia, a luz estruturada tem ganhado largas aplicações em muitos campos: inspeção em circuitos impressos, inspeção em manufaturas (defeitos em superfícies, saliências em objetos, detecção de bordas, medição de profundidade e alinhamento), e orientação de robôs [20].

Há uma grande quantidade de artigos abordando o uso da luz estruturada. Além de estabelecer um modelo preciso de visão, a calibração eficiente deste modelo é parte importante para visão *3D*. O objetivo de calibrar a luz estruturada é estabelecer uma relação prática de mapeamento entre pontos das coordenadas do mundo, onde a luz estruturada se encontra, e seus pontos correspondentes de projeção na imagem [21]. Um método de auto re-calibração baseado em plano de homografia é proposto em [22] enquanto que em [23] é mostrado um método de calibração e modelagem para múltiplos perfis estruturados de linha de luz.

Recentemente, uma outra técnica de luz estruturada tem crescido em importância. Esta técnica é baseada na codificação de uma luz projetada sobre a cena para ser usada como ferramenta para obtenção de informação de profundidade e reconstrução *3D*.

A Figura 2.10 mostra o uso desta técnica, proposta em [24], denominada *codificação temporal*. Cada coluna do padrão de luz pode estar iluminada ou não, podendo ser usadas várias máscaras para codificar qualquer ponto deste padrão em intervalos de tempos, com projeções seqüenciais de diferentes padrões.

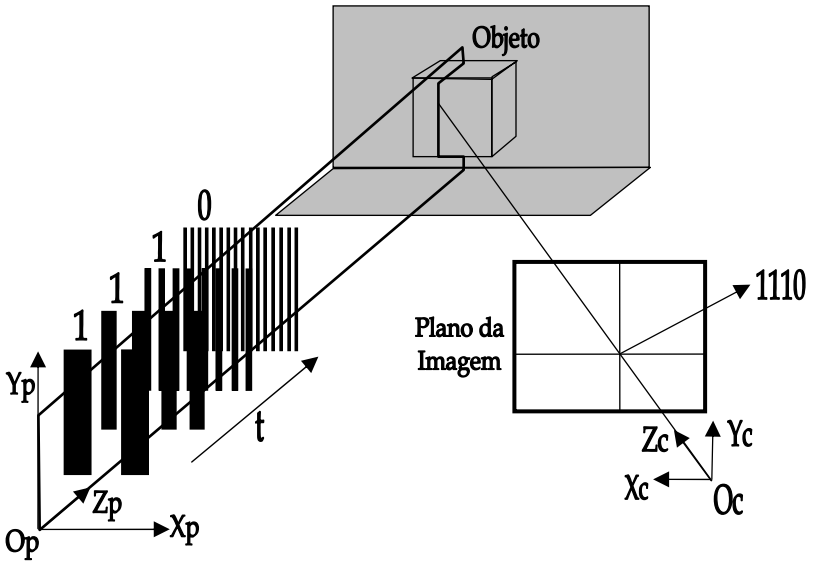


*Figura 2.10: Luz estruturada codificada sobre objeto.*

O número de padrões a serem projetados é determinado pelo número de colunas a serem codificadas. Então, cada posição da superfície a ser medida é temporal e seqüencialmente iluminada por valores diferentes para o mesmo ponto de análise na imagem. Isto produz um código que é distinto em relação a um ponto vizinho.

A câmera com o centro ótico $O_C$ é colocada a um certo ângulo do projetor, cujo referencial é $O_P$. A câmera armazena imagens em tons de cinza $\mathbf{I}(x, y, t)$ dos padrões distorcidos da luz, em diferentes tempos *t*. Para cada *pixel* na imagem, o código é armazenado. Com o uso deste código e dos parâmetros de orientação conhecidos através da calibração do sistema de aquisição, a informação *3D* da cena pode ser calculada.

A robustez da técnica de luz codificada depende da correta detecção de bordas entre as áreas iluminadas e não iluminadas. Usando parâmetros de calibração, a informação de profundidade pode ser calculada ao longo das bordas.

Em [25] é mostrada uma revisão de algumas técnicas para a codificação da luz, propostas por alguns autores. A resolução do problema da decodificação de um determinado ponto,

dentro de um padrão de luz estruturada é de grande importância. Em [26] é mostrado um algoritmo simples para resolver este problema.

## 2.3 Técnica de Medição para o Robô Quadrúpede Guará

O robô quadrúpede Guará foi desenvolvido como parte integrante das teses de doutorado [2] e [3], e em parceria criada entre o Programa de Pós-Graduação em Engenharia Elétrica da Ufes e a empresa Automatica Tecnologia S.A. Trata-se de um robô com uma plataforma e 4 pernas, cada qual com 4 graus de liberdade, permitindo, assim, trajetórias retilíneas e curvilíneas. A Figura 2.11 mostra o robô quadrúpede Guará.

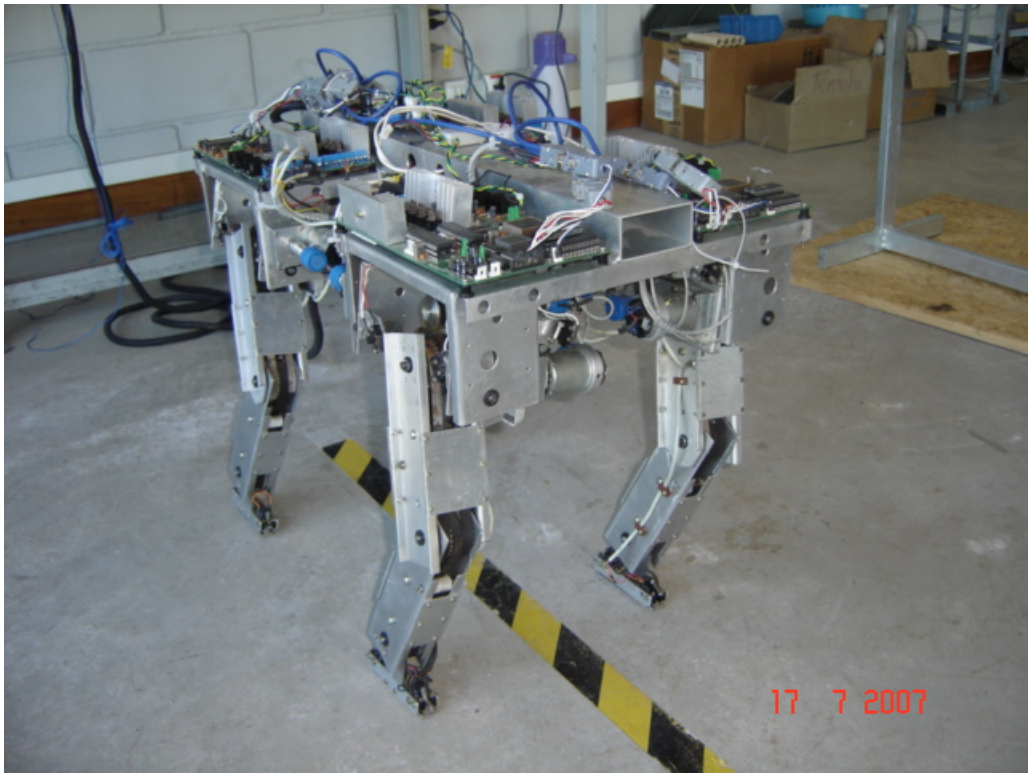


*Figura 2.11: O robô quadrúpede Guará, fonte [2].*

O robô permite navegação em ambientes semi-estruturados, [2] e [3] desenvolveram os modelos que são utilizados para odometria, controle e supervisão do robô Guará.

Para se locomover, o robô precisa saber onde pisar e acionar precisamente as patas, como manipuladores isolados, para posições de apoio ou de vôo para novo posicionamento.

Há duas instâncias de controle no robô. A primeira instância é responsável por coordenar o movimento das patas em função das restrições de equilíbrio e por definir as posições para as quais as patas devem ser acionadas, em função da orientação ou rota a ser seguida pelo robô. A segunda instância é responsável pelo acionamento da perna, como um manipulador, posicionando a pata, como um efetuador, na posição de apoio definida na primeira instância [2].

A arquitetura proposta do robô está moldada à estratégia de controle hierárquico, sendo sua estrutura definida para atender às seguintes condições:

- ✓ garantir o equilíbrio do robô durante sua locomoção em trajetórias com ou sem obstáculos, definindo a necessidade de observação contínua do estado atual real de toda a estrutura do robô, para que se decida a manutenção ou mudança do padrão dos movimentos subseqüentes [3];

- ✓ a interação do robô com o ambiente de navegação exige da arquitetura a capacidade de reação a qualquer restrição de movimentos que possam dar origem a desequilíbrio, desvios de trajetória ou danos físicos. [3].

O sistema sensorial de visão proposto para o robô Guará deve permitir que o robô reconheça e evite obstáculos que forem encontrados na sua trajetória, mapeando toda a superfície de forma precisa.

A plataforma do robô Guará pode parar por um instante, entre um passo completo e outro. Portanto, o sistema proposto deve ser acionado neste momento, para fazer o mapeamento da superfície.

O custo computacional do sistema proposto para o mapeamento da superfície deve ser levado em conta, haja vista que o sistema de controle do robô também está em execução. Ambos os sistemas estão em processos diferentes, mas na mesma unidade computacional.

A complexidade do algoritmo do sistema proposto é diretamente proporcional ao custo computacional. O sistema proposto deve ter uma implementação viável, com bons resultados, mas que não seja complexo demais para inviabilizar o uso na plataforma do robô Guará, demandando um tempo de processamento tão grande que comprometa a eficiência do sistema como um todo.

O peso do sistema proposto também é um fator a se considerar. O sistema deve ser leve, para não causar mudanças significativas no centro de gravidade e na estabilidade do robô.

A medição de distância por interferometria, embora precisa, não é recomendada neste sistema, devido ao fato de a medição de distância ser relativa a um referencial e sofrer muita influência das condições ambientais.

A medição por deslocamento de fase e a medição por tempo de vôo, ambas usando ondas eletromagnéticas ou laser, são recomendadas para distâncias maiores do que a altura da plataforma do robô.

A medição por tempo de vôo usando ondas sonoras é de fácil processamento, devido à natureza lenta do som, mas tem como desvantagem uma baixa resolução angular, resolução de faixa limitada e reflexões especulares.

Considerando as características previamente citadas, a técnica de medição definida para o robô Guará é a medição de distância por triangulação, devido à sua simplicidade de implementação.

Entre as técnicas de medição por triangulação, a visão estéreo, por se tratar de uma triangulação passiva, demanda boa iluminação ambiente, sua geometria epipolar tem o

problema da dificuldade de localizar os pontos correspondentes nas duas imagens e o algoritmo é por natureza complexo, demandando mais tempo computacional.

Outra técnica de medição por triangulação é o fluxo óptico. Contudo, esta técnica exige que haja um movimento relativo entre o robô e a superfície. O robô Guará é lento e pára alguns instantes quando muda para o próximo passo. Daí, esta técnica não seria viável para esta plataforma.

Finalmente, a medição por triangulação usando a luz estruturada tem características que a tornam a melhor opção para a plataforma do robô Guará. O robô, durante o momento em que ele se programa para o passo seguinte, pára, e a técnica entra em ação.

A técnica da luz estruturada com luz codificada, se comparada com a luz codificada usando laser e câmera *CCD*, demanda um processamento computacional maior, devido à decodificação da luz, além de o projetor ser um equipamento com um peso significativo para se colocar na plataforma do robô.

A técnica da luz estruturada com laser e câmera *CCD* tem um custo computacional mais baixo, obtendo-se, através da mudança do perfil da linha de laser sobre os objetos, um vasto conjunto de dados. A linha de laser *varre* toda a superfície e a câmera *CCD* adquire imagens em quadros sucessivos. Posteriormente através destes vários perfis são extraídas informações de profundidade dos objetos, permitindo a geração de imagens de profundidade. Estas imagens são fundamentais para o mapeamento da superfície.

A Figura 2.12 mostra o robô quadrúpede Guará, com o sistema proposto instalado à sua frente.

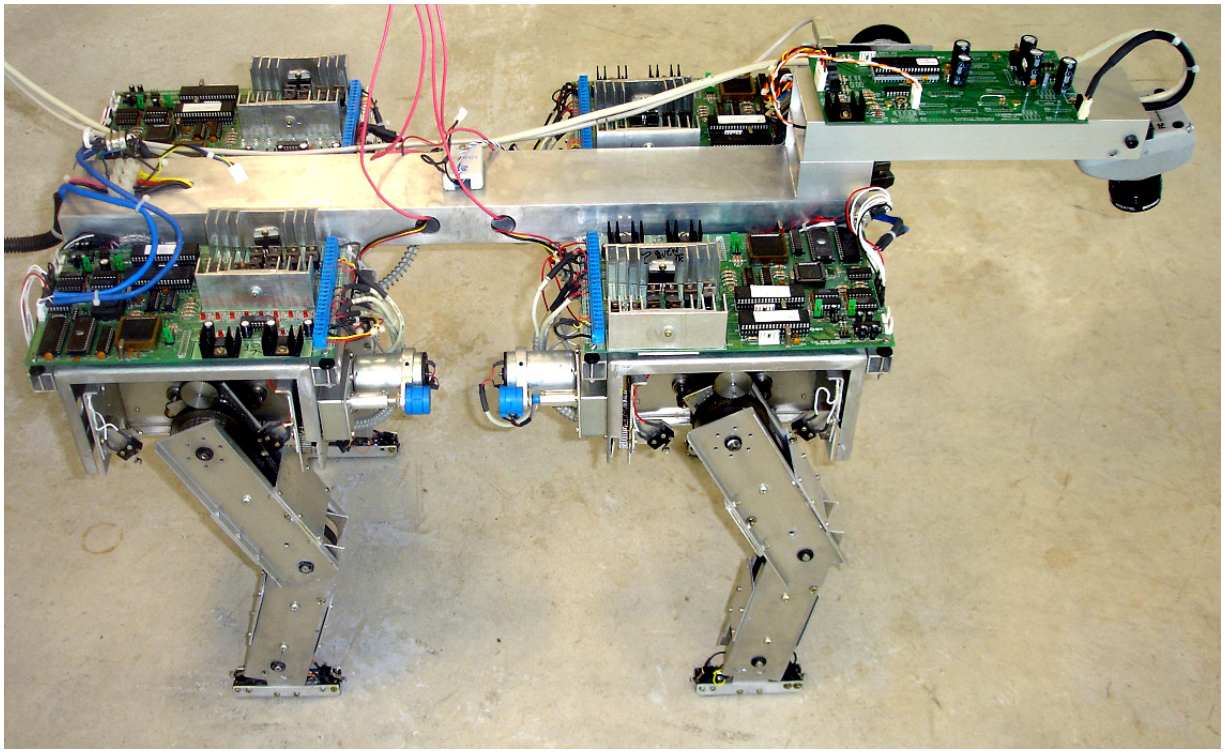

*Figura 2.12: O robô quadrúpede Guará com o sistema proposto instalado à frente.*

# Capítulo 3 - Arquitetura e Modelagem do Sistema de Visão

## 3.1 Introdução

Para a implementação da técnica de triangulação ativa com luz estruturada, com estabilidade adequada para uso na plataforma do robô Guará, um sistema baseado em uma linha de laser de baixa potência e uma câmera *CCD* foi implementado.

O sistema proposto contém uma placa microcontrolada que permite a comunicação serial, via *USB*, com o computador. O laser é direcionado por um espelho rotativo, fixado ao eixo de um motor de passo. O circuito de controle do motor de passo, fontes de tensão, e o sensor de posição de início de quadro, estão contidos nesta placa.

Ao receber um sinal inicial de controle do computador, o motor de passo é ligado e o laser é trazido para o início de aquisição do quadro. A cada sinal de aquisição, o laser anda um, dois ou quatro passos, em função do tipo de varredura escolhido, e uma imagem é adquirida. O processo se repete até que toda a superfície seja varrida. Há 3 modos de varredura: grossa, padrão e fina, que correspondem ao avanço de quatro, dois e um passo do motor de passo, respectivamente.

As imagens adquiridas são tratadas por vários algoritmos, extraindo somente o perfil do laser, somando todas as imagens em uma única imagem, e gerando a imagem de profundidade do ambiente. Mesmo com a simplicidade construtiva intrínseca, as imagens de profundidade obtidas por este sistema são precisas e adequadas para o uso no robô.

Dentro desta concepção de luz estruturada com laser, muitos sistemas referenciados na literatura foram estudados. Um sistema de medição usando laser de baixa potência para robôs autônomos é proposto em [27] e [28]. Em [29] é mostrado um sistema com laser não

nocivo para os olhos em triangulação com uma câmera *CCD*, com bom campo de visão e baixo custo. Este sistema usa laser classe I.

Um sistema de medição para visão de robô com luz estruturada codificada, compacto e de alta velocidade é proposto em [30]. O princípio de medição de distância é baseado em um método de decodificação espacial. Outra referência que usa este mesmo princípio, descrevendo um sistema preciso de medição é [31].

Há também uma série de medidores de distância (*rangefinders*) comerciais, baseados em laser. Entre eles, as empresas *Acuity*, *Sick*, *Riegl* e *Cyra* têm vários modelos. Há em [32] uma comparação entre os modelos destes fabricantes, discriminando as principais características, além da descrição de um medidor a laser de baixa potência e compacto. Uma boa revisão dos sistemas de medição *3D* a laser é feita em [33].

Uma solução de medidor de distância para captura *3D* de objetos, com baixo custo e fácil configuração, é descrita em [34].

A vantagem da arquitetura proposta é unir recursos necessários para que imagens de profundidade sejam geradas com precisão e com baixo custo computacional. O sistema é leve, simples e compacto. Ele é instalado na plataforma do robô, sem causar instabilidade ao seu funcionamento.

## 3.2 Arquitetura do Sistema de Visão

O Sistema de visão a laser está fundamentado numa arquitetura formada por 4 módulos funcionais interligados, cada qual responsável por uma parte específica do controle do sistema, denominados:

a) módulo de Supervisão e Controle;
b) módulo de Controle do Laser;
c) módulo de Tratamento de Imagens;

d) módulo de Mapeamento da Superfície.

Estes módulos são detalhados nos itens subseqüentes. A Figura 3.1 mostra um diagrama de blocos dos 4 módulos funcionais, e a interligação com o sistema de controle do robô, o qual é tratado em [2] e [3].

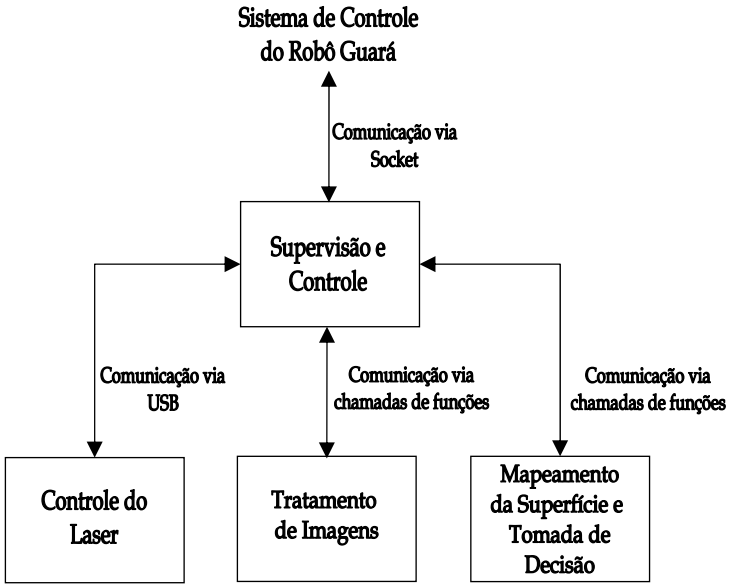


*Figura 3.1: Diagrama de blocos dos módulos funcionais do sistema de visão a laser.*

A Figura 3.2 mostra um desenho de montagem do sistema de visão a laser. Este sistema é fixado na frente do robô Guará, com o laser posicionado de maneira a varrer todo o campo de visão da câmera *CCD*. Há, na saída do canhão do laser, uma lente que espalha o laser em forma de uma linha retilínea e fina, a qual é visível na imagem capturada pela câmera.

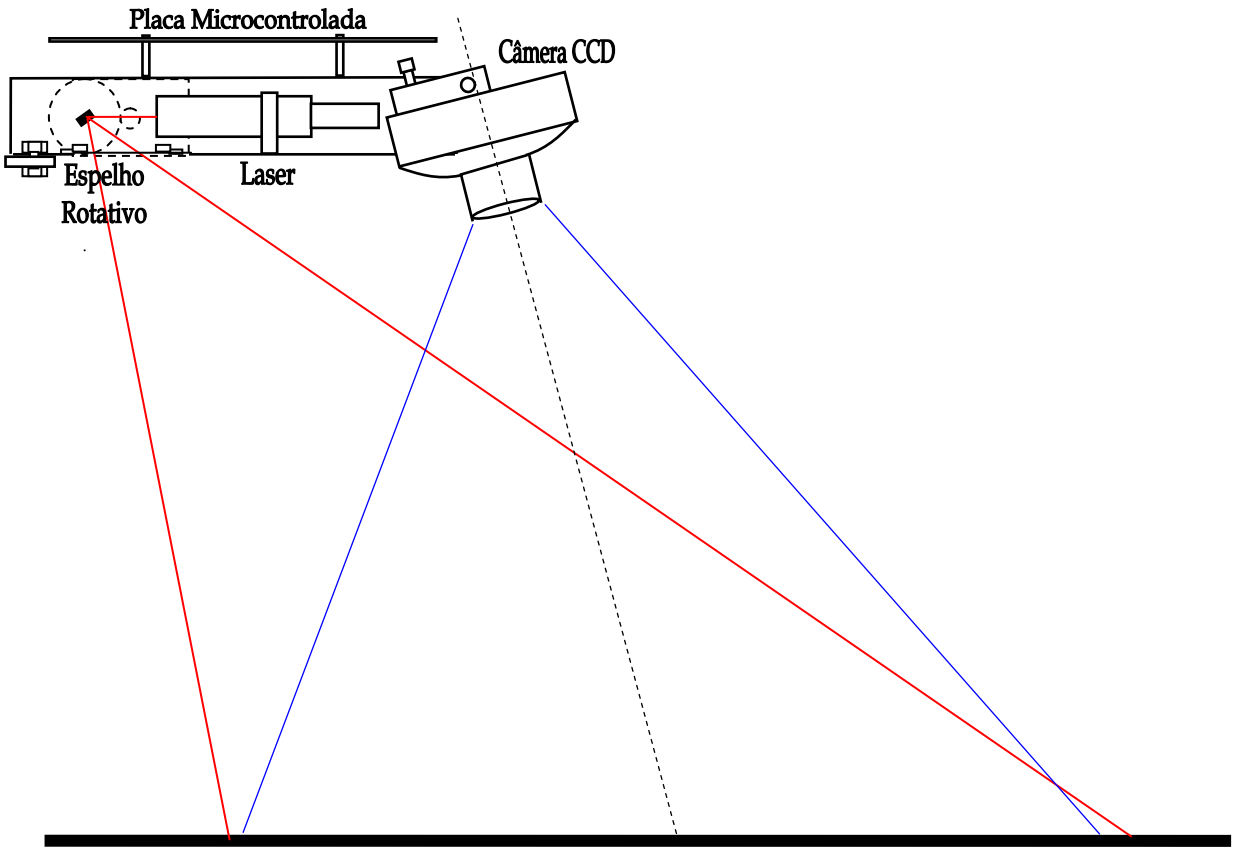


*Figura 3.2: Desenho de montagem do sistema de visão a laser.*

### 3.2.1 Módulo de Supervisão e Controle

Este módulo é uma aplicação na plataforma *.Net*, desenvolvida em ambiente integrado, usando ferramenta de programação avançada (*Visual Studio .Net C#*), que permite o controle total do sistema e a interface com o usuário. As principais funcionalidades deste módulo são:

- ✓ iniciação e término do sistema;
- ✓ temporização de rotinas;
- ✓ comunicação com o sistema de controle do robô,
- ✓ comunicação e iniciação dos outros módulos;
- ✓ sincronização do laser com a câmera *CCD*;
- ✓ aquisição de imagem em estado original;
- ✓ escolha de resolução do laser;

- ✓ calibração do módulo de controle do laser;
- ✓ calibração de parâmetros da câmera *CCD*;
- ✓ configuração de parâmetros da superfície;
- ✓ configuração dos referenciais da câmera e do robô;
- ✓ gravação e recuperação das configurações do sistema por arquivos.

### 3.2.2 Módulo de Controle do Laser

Este módulo microcontrolado controla o movimento do laser através de um conjunto motor de passo com espelho rotativo. Ele é responsável por gerar um padrão de luz estruturada sobre a superfície de navegação do robô, com a incidência de múltiplas linhas de laser, com 3 níveis de resolução: grossa, padrão e fina. As principais funcionalidades são enumeradas a seguir:

- ✓ comunicação serial (*USB*) com o módulo de supervisão e controle;
- ✓ posicionamento do laser no início da imagem;
- ✓ controle do passo do laser de avanço e retorno;
- ✓ sincronização com o módulo de tratamento de imagens;
- ✓ detecção de sensor do início do curso do laser;
- ✓ varredura do laser sobre a superfície com a resolução escolhida.

### 3.2.3 Módulo de Tratamento de Imagens

Este módulo executa um papel relevante no sistema. Ele é dotado de uma Câmera *CCD* usando a interface serial padrão *Firewire* (*IEEE 1394*), cujas imagens são obtidas por interrupção através do *driver DirectX*© do sistema operacional *Windows*©, dentro da aplicação em tempo de execução.

As imagens adquiridas pelo módulo de supervisão e controle são sincronizadas com o laser, e uma vez armazenadas necessitam de tratamento, para posterior geração de imagens de profundidade. A seguir, as principais características deste sistema são enumeradas:

- ✓ sincronização com a aquisição da imagem;
- ✓ armazenamento de imagens originais;

- ✓ execução de vários algoritmos de tratamento de imagens;
- ✓ geração de imagens de profundidade (*range images*);
- ✓ geração de imagens de profundidade orto-retificadas;
- ✓ registro de imagens de profundidade sucessivas;
- ✓ extração de descritores das imagens de profundidade.

### 3.2.4 Módulo de Mapeamento da Superfície e Tomada de Decisão

Trata-se do módulo chave do sistema de visão a laser. Ele é responsável pela identificação e mapeamento dos obstáculos da superfície. Seu objetivo é mapear a superfície em função dos referenciais do robô e do sistema proposto.

A metodologia de navegação reativa usando árvore binária de decisão é implementada para que o robô possa navegar em uma superfície com obstáculos.

A seguir, as principais características deste módulo são enumeradas:

- ✓ aquisição de informação de odometria do robô;
- ✓ aquisição de informação da cinemática do robô;
- ✓ aquisição de coordenadas do robô e das patas;
- ✓ aquisição de descritores dos obstáculos da superfície;
- ✓ execução de algoritmo para mapeamento da superfície;
- ✓ execução de algoritmo de controle para tomada de decisão baseado em navegação reativa.

## 3.3 Modelagem do Sistema de Visão

O robô Guará, concebido originalmente sem sistema de visão, navega por odometria. Uma câmera *CCD*, juntamente com um sistema microcontrolado com laser rotatório, geram imagens da superfície, que necessitam ser acondicionada em vários aspectos, para que se possa obter informações que garantam a navegabilidade do robô.

O tratamento de imagem é feito através de vários algoritmos, abrangendo desde a conversão da imagem para tons de cinza até geração da imagem de profundidade, agrupados e otimizados no código, para que tenham um custo computacional menor. Desta maneira obtém-se, após uma calibração do sistema, um mapeamento preciso de cada ponto no espaço *3D* da superfície através do plano *2D*.

O mapeamento *2D* da imagem não permite uma reconstrução *3D*. Com o uso de laser, gerando um padrão de luz estruturada na superfície, são obtidas as cotas de profundidade dos obstáculos, que, uma vez sincronizadas, geram imagens de profundidade (*range images*).

As imagens de profundidade são verdadeiros mapas de superfície, onde ao valor de intensidade $f(x, y)$ de um ponto $(x, y)$ da imagem é atribuída uma profundidade, que pode ser quantificada como um valor de cinza, permitindo o seu armazenamento no formato de imagem. Imagens de profundidade, também conhecidas como mapas de profundidade, são largamente abordadas na literatura, referenciadas em [35] e [36].

## 3.4 Transformação de Referenciais da Imagem para a Câmera

Esta transformação permite que pontos *3D* da superfície real à frente do robô sejam transformados em pontos *3D* no sistema de referenciais da câmera *CCD*, cujo centro do referencial é o centro ótico da câmera. Esta transformação é dividida em duas partes:

a) **Obtenção das coordenadas (*x,y)* do plano da superfície**: através das imagens adquiridas, usando-se o modelo da câmera pontual e com vários algoritmos de tratamento de imagens, além de correção da distorção radial, transformação projetiva e projeção ortogonal;

b) **Obtenção da coordenada *z* do plano da superfície**: através da incidência de linhas de laser, usando-se uma modelagem matemática, para cada ponto *(x,y)* determina-se a coordenada *z* (profundidade do obstáculo).

### 3.4.1 O Modelo de Câmera

O modelo de câmera amplamente aplicado em visão computacional é o modelo pontual (*pinhole model*). As câmeras em perspectiva baseiam-se neste princípio. Os raios luminosos vindos de um objeto passam por um pequeno orifício na entrada da câmera e são projetados no plano da imagem [37]. Esta representação, embora simples, é bastante funcional para modelar câmeras. A Figura 3.3 mostra este modelo.

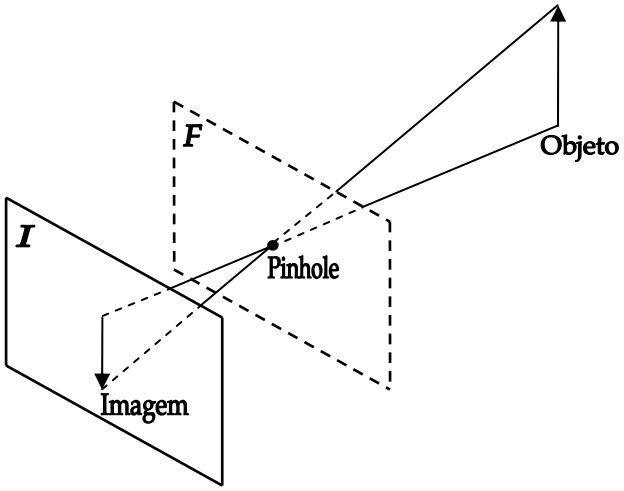


*Figura 3.3: O modelo pontual de câmera.*

O plano *I* é o plano da imagem, que é o plano onde a imagem é formada por projeção. O plano *F* é o plano focal, paralelo ao plano da imagem e fica localizado a uma distância *f* do mesmo, que corresponde à distância focal. A Figura 3.4 mostra o sistema de coordenadas da câmera. O ponto *c* é o centro óptico, e sua distância ao plano *I* é a distância focal *f*. Pela projeção, um objeto representado pontualmente por *n* no espaço *3D* é formado pela interseção da reta <*c,n*> com o plano *I*.

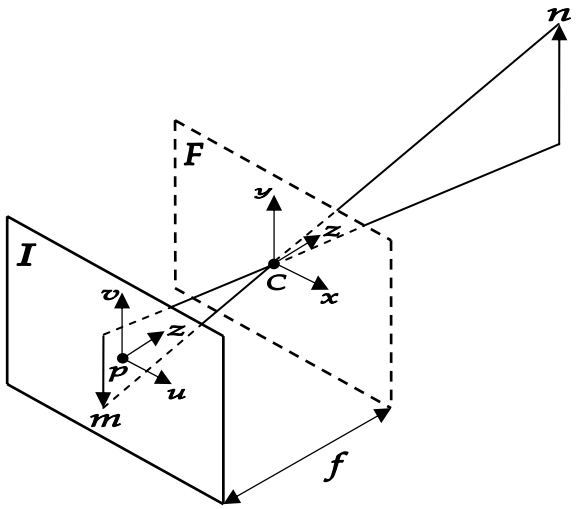


*Figura 3.4: O sistema de coordenadas da câmera.*

Considere o sistema de coordenadas da câmera no plano *F,* que tem como origem o ponto *c* e o seu eixo *Z* ao longo do eixo óptico, e as coordenadas do plano de imagem *I* que têm como origem o ponto *p*. Um ponto *n*, cujas coordenadas representadas no referencial da câmera são $(x, y, z)$, tem uma projeção *m* no plano da imagem *I,* cujas coordenadas são representadas por $(u, v)$.

Para facilitar a análise geométrica, a Figura 3.4 é redesenhada, colocando-se o plano *F* da câmera como referencial e o plano da imagem *I* à sua frente, além de colocar o sistema de coordenadas do ponto *m* e *n* no mesmo sentido. A Figura 3.5 mostra as modificações, e por semelhança de triângulos, é possível mostrar a relação entre as coordenadas de *m* e *n* pelos referenciais adotados. Desta forma, obtém-se:

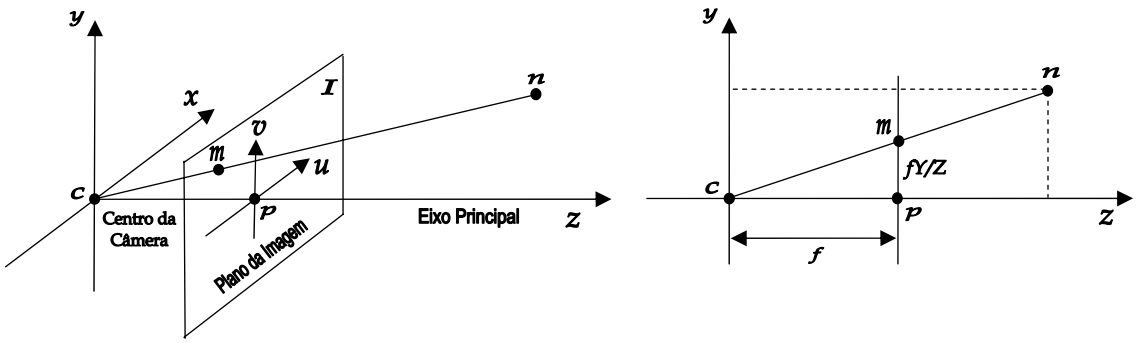


*Figura 3.5: Relação entre as coordenadas de m e n.*

$$\tilde{\mathbf{n}} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T \tag{3.1}$$

$$u = f.x/z \tag{3.2}$$

$$v = f.y/z \tag{3.3}$$

$$\tilde{\mathbf{m}} = \begin{bmatrix} u & v & 1 \end{bmatrix}^T = \begin{bmatrix} f.x/z & f.y/z & 1 \end{bmatrix}^T. \tag{3.4}$$

Onde $\tilde{\mathbf{m}}$ e $\tilde{\mathbf{n}}$ são vetores homogêneos dos pontos *m* e *n,* respectivamente. Representando-se através de forma matricial, pode-se escrever uma relação linear através de coordenadas homogêneas, ou seja

$$\begin{bmatrix} f.x \\ f.y \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{3.5}$$

e logo

$$\mathbf{P} = diag(f, f, 1)\begin{bmatrix} \mathbf{I} \mid \mathbf{0} \end{bmatrix}. \tag{3.6}$$

Portanto, a sua representação matricial é

$$\tilde{\mathbf{m}} = \mathbf{P}.\tilde{\mathbf{n}}, \tag{3.7}$$

e $\mathbf{P}$ é conhecida como *matriz de projeção perspectiva*.

É importante notar que, usando a geometria projetiva, a projeção por perspectiva é descrita por uma equação linear, simplificando a abordagem, pois ao invés de usar equações não-lineares, como as equações (3.2) e (3.3), usa-se uma relação linear. Pode-se afirmar,

portanto, que uma câmera constitui de um sistema capaz de realizar uma transformação linear do espaço projetivo $P^3$ para o plano projetivo $P^2$.

A equação (3.5) considera que a origem do sistema de coordenadas do plano de imagem *I* é o ponto principal *p*. Contudo, esta suposição nem sempre é adotada, pois geralmente adota-se o canto da imagem como a origem do referencial do plano da imagem, conforme mostrado na Figura 3.6.

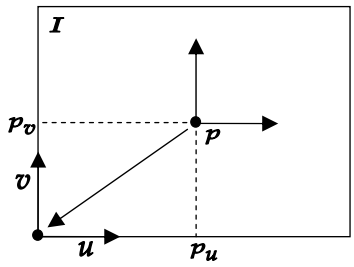


*Figura 3.6: Mudança de referencial do plano da imagem.*

Assim, $(p_u, p_v)$ é o ponto principal do plano da imagem. Logo, o ponto *m* no novo referencial é denominado $m_n$, e é dado por

$$m_n = (u + p_u, v + p_v), \tag{3.8}$$

cujo vetor de coordenadas homogêneas é

$$\tilde{\mathbf{m}}_\mathbf{n} = \begin{bmatrix} f.x + z.p_u & f.y + z.p_v & 1 \end{bmatrix}^T. \tag{3.9}$$

Portanto, reescrevendo a equação (3.5) tem-se que

$$\begin{bmatrix} f.x + z.p_u \\ f.y + z.p_v \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & p_u & 0 \\ 0 & f & p_v & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A} & | & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{3.10}$$

sendo

$$\mathbf{A} = \begin{bmatrix} f & 0 & p_u \\ 0 & f & p_v \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.11}$$

A análise do modelo da câmera até o presente momento foi baseada na geometria projetiva no espaço euclidiano de coordenadas, que assume que há escalas iguais em ambos os eixos axiais. Para uma câmera *CCD*, devido aos seus detalhes construtivos, as coordenadas são expressas em *pixel* (*picture element*), cujas escalas de seus eixos axiais podem ser diferentes e $m_u$ e $m_v$ representam o número de *pixels* por unidade de distância nos eixos *U* e *V*, respectivamente. Logo,

$$\mathbf{A} = \begin{bmatrix} m_u & 0 & 0 \\ 0 & m_v & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & p_u \\ 0 & f & p_v \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.12}$$

ou seja

$$\mathbf{A} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.13}$$

O par de coordenadas $(u_0, v_0)$ representa o ponto principal em *pixels*. Os valores $\alpha_u = m_u.f$ e $\alpha_v = m_v.f$ representam os fatores de escala nos eixos *U* e *V*, respectivamente. A matriz de *pixels* de uma câmera pode ter as linhas e colunas não perpendiculares. Neste caso, há a inclusão do parâmetro *s* (*skew*) que representa o fator de não perpendicularidade da matriz de *pixels*, conforme mostrado na Figura 3.7, que é dado por

$$s = \tan\gamma, \tag{3.14}$$

e, neste caso,

$$\mathbf{A} = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.15}$$

A matriz de transformação afim representada por **A** em (3.15) é a *matriz de calibração da câmera CCD,* e contém 5 parâmetros intrínsecos. Para a maioria das câmeras *CCD*, assume-se que os *pixels* são quadrados e fatores de escala $\alpha_u = \alpha_v$. Além do mais, o parâmetro *s* pode ser assumido como zero para fins práticos.

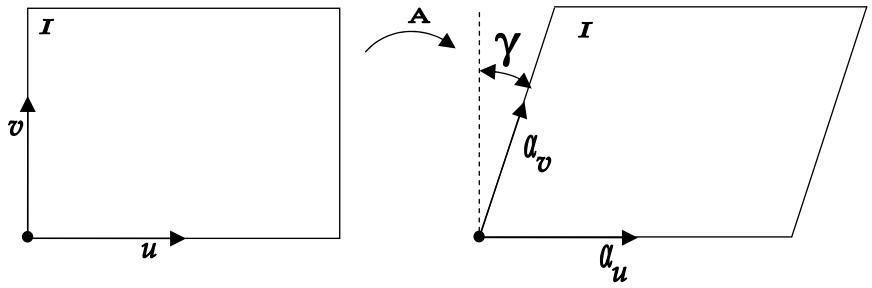


*Figura 3.7: Correção pelo fator de não perpendicularidade s.*

Um modelo de câmera pontual é descrito em [38] e [39], mostrando a matriz de calibração da câmera com os 5 parâmetros intrínsecos.

### 3.4.2 Distorção Radial

Imagens obtidas a distâncias curtas por câmeras com lentes grande-angulares, que é o caso típico deste sistema, exibem distorção significativa, especialmente distorção radial. A distorção radial, também conhecida como *efeito barril* ou *aberração esférica,* é modelada para estabelecer uma relação entre *pixels* com distorção e *pixels* sem distorção. A equação a seguir expressa matematicamente a distorção radial, com referência em [38] e [39], ou seja,

$$\begin{bmatrix} u_r \\ v_r \end{bmatrix} = \begin{bmatrix} u(1+\sum_{i=1}^{n} k_i r^{2i}) \\ v(1+\sum_{i=1}^{n} k_i r^{2i}) \end{bmatrix}, \tag{3.16}$$

onde $(u_r, v_r)$ é a coordenada do *pixel* distorcido e $(u, v)$ é a coordenada do *pixel* sem distorção, $r = \sqrt{u^2 + v^2}$ é o raio e $k_1, k_2, \ldots k_n$ são os coeficientes da distorção radial. Todas as coordenadas são referenciadas ao ponto principal da imagem $(u_0, v_0)$. Desconsiderando os termos de alta ordem, por eles terem pouca influência e considerando $k = k_1$, a equação (3.16) pode ser simplificada e re-escrita como

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \dfrac{u_r}{(1+kr^2)} \\ \dfrac{v_r}{(1+kr^2)} \end{bmatrix}. \tag{3.17}$$

Contudo, a equação (3.17) denota uma complexidade na solução. Desta forma, para efeitos práticos e devido ao fato de a distorção radial ser pequena, considera-se que $r = \sqrt{u^2 + v^2} \cong \sqrt{u_r{}^2 + v_r{}^2}$ [40], e logo

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \dfrac{u_r}{1 + k(u_r{}^2 + v_r{}^2)} \\ \dfrac{v_r}{1 + k(u_r{}^2 + v_r{}^2)} \end{bmatrix}. \tag{3.18}$$

O parâmetro *k* é medido de forma experimental. É colocado um plano de calibração quadriculado e a imagem obtida é analisada. À medida que o valor de *k* é incrementado, o efeito da distorção é reduzido na imagem, até a obtenção de um valor que neutraliza a distorção radial.

### 3.4.3 Geometria Projetiva

A representação homogênea para pontos *2D* fornece uma notação consistente para mapeamentos projetivos e afins. Ela foi usada em geometria projetiva muito antes da introdução da computação gráfica [41].

Na geometria euclidiana, os pontos são representados no plano $R^2$ por vetores $(x, y)$. A geometria projetiva usa o plano projetivo, um conjunto do plano real, cujas coordenadas homogêneas são $(x', y', a)$.

Na geometria projetiva, um ponto real *2D* $(x, y)$ é representado por um vetor homogêneo $\tilde{\mathbf{p}} = \begin{bmatrix} x' & y' & a \end{bmatrix} = \begin{bmatrix} a.x & a.y & a \end{bmatrix}$ onde $a$ é um valor arbitrário não nulo. Geralmente usa-se $a = 1$.

Para se recuperar as coordenadas reais de um vetor homogêneo, basta dividi-las pelo componente homogêneo $a$, logo $(x, y) = (x'/a, y'/a)$. O espaço projetivo também inclui os *pontos no infinito* formados por vetores do tipo $(x', y', 0)$, exceto a origem $(0,0,0)$. Os *pontos no infinito* são pertencentes à *reta do infinito*.

A Figura 3.8 mostra a projeção de um ponto $n$, pertencente ao plano *R,* em $m$, que pertence ao plano *I*. Quatro pontos conhecidos da superfície são associados a quatro pontos conhecidos da imagem, neste caso os quatro cantos da imagem.

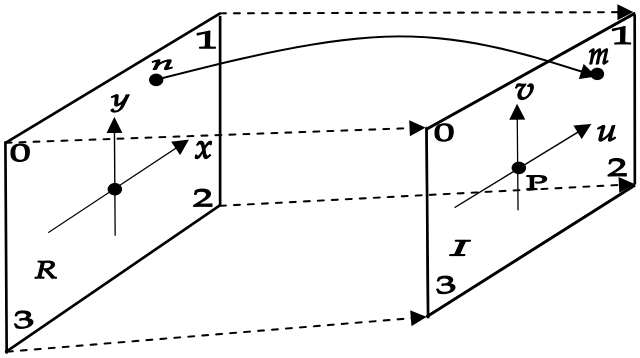


*Figura 3.8: Transformação projetiva: projeção de um ponto $n$ pertencente ao plano R em $m$ que pertence ao plano I.*

A transformação projetiva é uma projeção de um plano, através de um ponto, em outro plano. Elas são muito usadas em transformações homogêneas de câmera, em que um ponto do plano real é mapeado em um ponto do plano da câmera.

Uma importante propriedade da transformação projetiva é que a *inversa de uma transformação projetiva é uma transformação projetiva*. Portanto, é possível mapear $n$ em $m$ e vice-versa.

A transformação projetiva é um mapeamento dado por [41]

$$x = \frac{au + bv + c}{gu + hv + i}, \tag{3.19}$$

$$y = \frac{du + ev + f}{gu + hv + i}, \tag{3.20}$$

onde $x$ e $y$ são as coordenadas do plano *R,* enquanto que $u$ e $v$ são as coordenadas do plano da imagem *I*.

A manipulação da transformação projetiva é mais fácil na notação de matriz homogênea, pois é uma transformação linear. Considerando-se o plano *I*, onde $p_s = (u', v', q)$ representa o ponto de origem e o plano R, onde $p_d = (x', y', a)$ representa o ponto de destino, obtém-se que

$$\tilde{\mathbf{p}}_d = \mathbf{M}.\tilde{\mathbf{p}}_s \tag{3.21}$$

e

$$\begin{bmatrix} x' \\ y' \\ a \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} . \begin{bmatrix} u' \\ v' \\ q \end{bmatrix}, \tag{3.22}$$

onde $(x, y) = (x'/a, y'/a)$ para $a \neq 0$ e $(u, v) = (u'/q, v'/q)$ para $q \neq 0$. Portanto há 9 coeficientes a serem determinados usando eliminação Gaussiana. Contudo pode-se assumir que $i = 1$, exceto no caso especial em que o ponto de origem $(0,0)$ mapeia um ponto no infinito. Desta forma, 8 coeficientes ( $a - h$ ) são calculados pela eliminação Gaussiana.

Um caso particular da transformação projetiva é quando $g = h = 0$, ou seja, uma linha no infinito é mapeada nela própria, tratando-se, neste caso, de uma Transformação Afim.

A transformação projetiva é muito útil para corrigir alterações na imagem de uma câmera devido ao efeito da perspectiva, que ocorre quando o plano da imagem não é paralelo ao plano real da superfície, além de corrigir qualquer não alinhamento entre os planos da superfície e da câmera. Esta transformação, portanto, é usada na calibração da câmera do sistema de aquisição e tratamento de imagens proposto neste trabalho.

A Figura 3.9 mostra geometricamente o processo da transformação, em que o ponto de visão, representado pelo centro ótico *C*, é deslocado para o ponto de visão *C'*, de modo que o plano da imagem corrigida fica perpendicular à superfície.

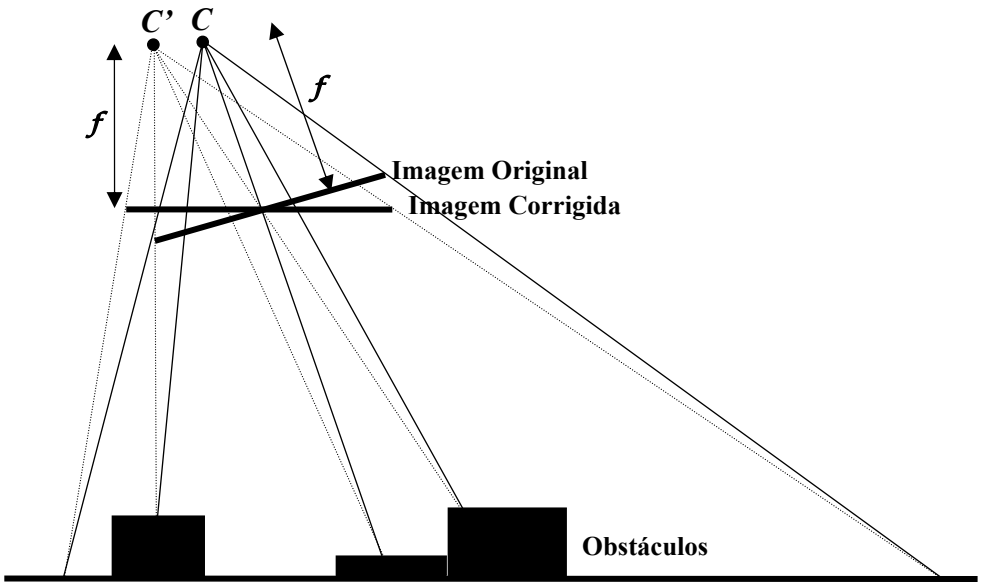


*Figura 3.9: Centro ótico deslocado após a transformação projetiva.*

Para viabilizar esta transformação, é necessário especificar 4 pontos do quadrilátero de origem (plano da superfície) e seus 4 pontos correspondentes no quadrilátero de destino (plano da imagem). Estes pontos de origem são considerados *marcas fiduciais* da imagem em perspectiva, e são associados aos 4 cantos da imagem transformada, de modo a se ter um mapeamento que abranja toda a sua área.

As *marcas fiduciais* são obtidas através de *click* do *mouse*, durante a calibração da câmera, usando um plano de calibração em formato de tabuleiro de xadrez colocado alinhado ao campo de visão da câmera.

A Figura 3.10 mostra o processo da transformação projetiva. Os pontos 0, 1, 2 e 3 da imagem de origem são associados aos 4 cantos da imagem de destino, mapeando a superfície.

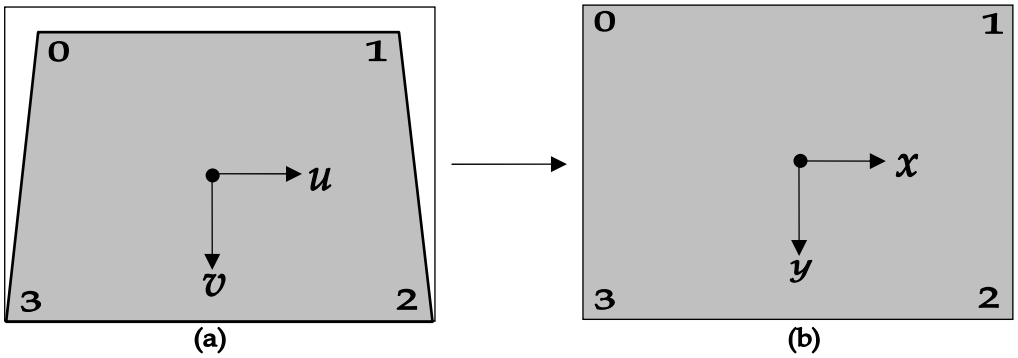


*Figura 3.10: Transformação projetiva para correção de imagem, (a) imagem em perspectiva inclinada e (b) imagem corrigida associada à superfície.*

Na transformação projetiva, 8 coeficientes devem ser determinados, em função dos pontos de origem e destino das imagens. O mapeamento correspondente $(u_k, v_k)$ para $(x_k, y_k)$, onde $k = 0,1,2,3$, é realizado com base nas equações (3.19) e (3.20), e considerando que todas as coordenadas são finitas, logo $i = 1$ [41]. Para determinar os 8 coeficientes $a - h$ desconhecidos são necessárias 8 equações, como segue:

$$x_k = \frac{au_k + bv_k + c}{gu_k + hv_k + 1} \Rightarrow u_k a + v_k b + c - u_k x_k g - v_k x_k h = x_k \tag{3.23}$$

$$y_k = \frac{du_k + ev_k + f}{gu_k + hv_k + 1} \Rightarrow u_k d + v_k e + f - u_k y_k g - v_k y_k h = y_k. \tag{3.24}$$

Para $k = 0,1,2,3$, as equações anteriores podem ser escritas, na forma matricial, como

$$\begin{bmatrix} u_0 & v_0 & 1 & 0 & 0 & 0 & -u_0x_0 & -v_0x_0 \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3x_3 & -v_3x_3 \\ 0 & 0 & 0 & u_0 & v_0 & 1 & -u_0y_0 & -v_0y_0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1y_1 & -v_1y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2y_2 & -v_2y_2 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -u_3y_3 & -v_3y_3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}. \tag{3.25}$$

O sistema linear 8x8 é resolvido usando eliminação Gaussiana, determinando assim os 8 coeficientes $a - h$ usados na calibração da câmera *CCD*.

A Figura 3.11 mostra a imagem original do plano de calibração quadriculado e a imagem transformada já com a correção da distorção radial, obtidas pelo sistema. Após a transformação, cada marca preta ou branca ocupa uma área de 30x30 $mm^2$.

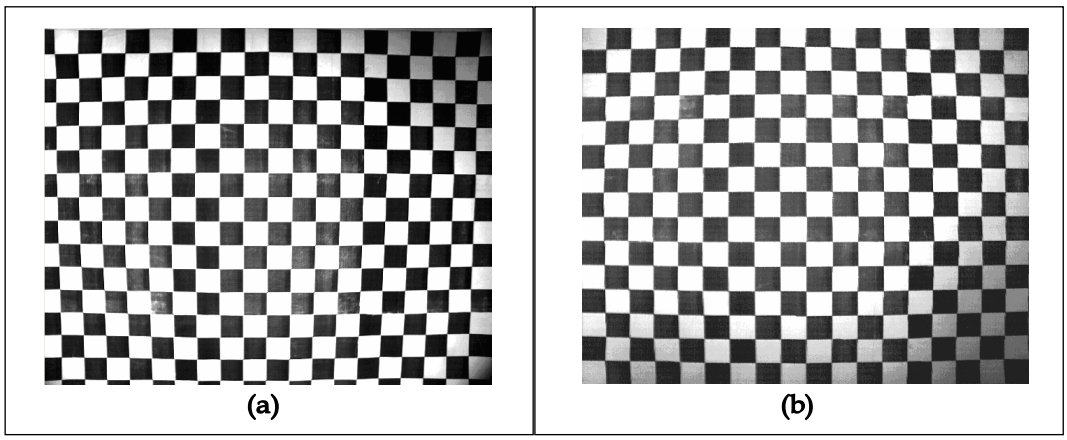


*Figura 3.11: (a) Imagem original (b) Imagem sem distorção radial e corrigida pela transformação projetiva.*

A Figura 3.12 mostra uma superfície com 5 objetos de alturas diferentes de uma imagem original e os mesmos 5 objetos em uma imagem transformada. Para preencher os *pixels* não mapeados da imagem transformada, é feita uma interpolação bi-linear. Observe-se que as 4 marcas fiduciais da superfície são associadas aos 4 cantos da imagem transformada.

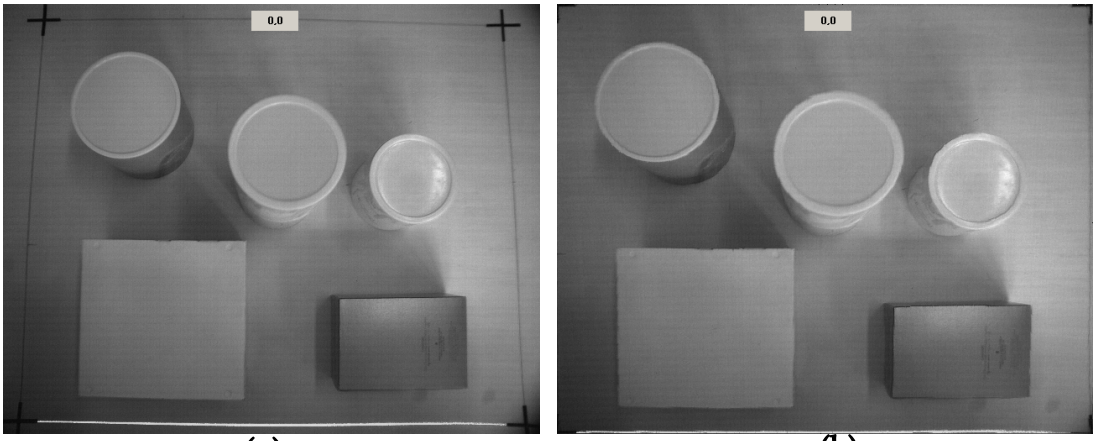


*Figura 3.12: (a) Imagem original (b) Imagem sem distorção radial e corrigida pela transformação projetiva em tons de cinza.*

### 3.4.4 Informação de Profundidade

Através da transformação projetiva, um mapeamento de um par ordenado $(x, y)$ da superfície é associado a um par ordenado $(u, v)$ no plano da imagem. Contudo, a informação de profundidade dos objetos não é obtida.

Pela triangulação, na prática, o ângulo de inclinação do eixo do laser em relação ao eixo ótico da câmera é difícil de medir com precisão. Um método usando equação da reta é proposto em [42], facilitando a modelagem para incidência de múltiplas linhas de laser, que é o caso do sistema proposto.

Na Figura 3.13, para um objeto 1 com altura $h_1$, cuja incidência do laser é no ponto $p_1 = (x_1, y_1, z_1)$, obtém-se no plano da imagem a cota $v_1$. De forma análoga, um objeto 2 com altura $h_2$, cuja incidência do laser é no ponto $p_2 = (x_2, y_2, z_2)$, obtém-se no plano da imagem a cota $v_2$. As alturas dos objetos 1 e 2 são conhecidas durante o processo de calibração.

O sistema da Figura 3.13 mostra que o eixo ótico da câmera não está perpendicular ao plano da superfície.

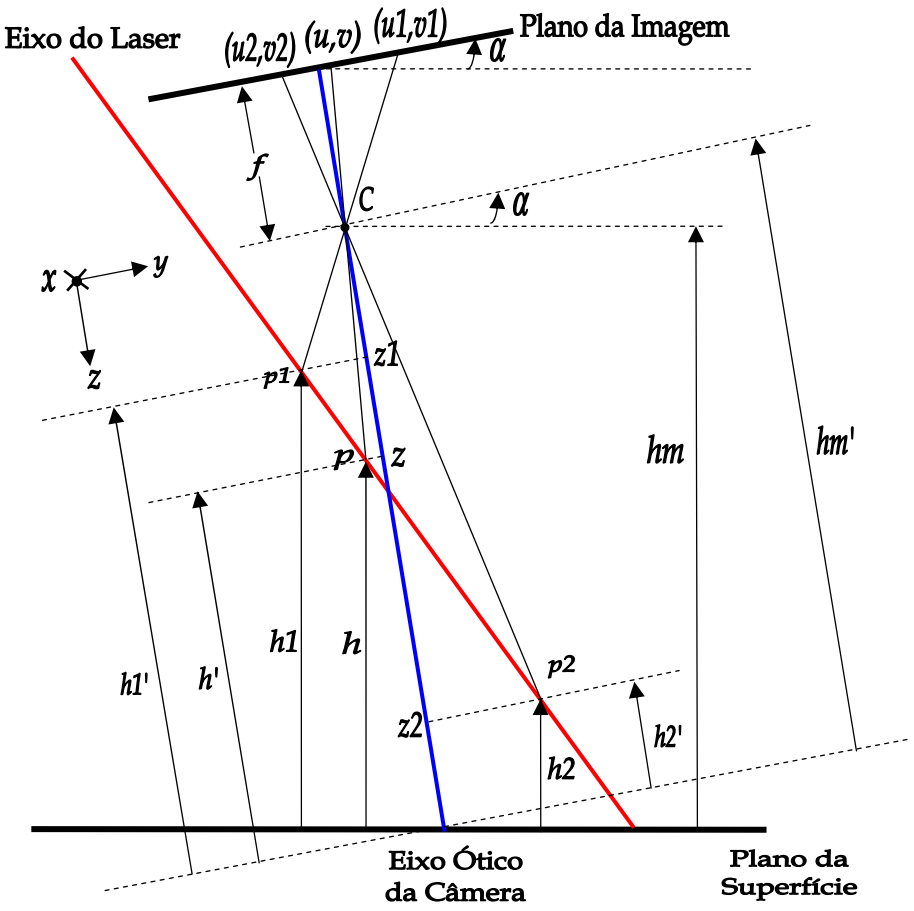


*Figura 3.13: Sistema de eixos laser-câmera e a representação dos objetos no plano da superfície e da imagem.*

Da Figura 3.13, obtém-se que: $z = h'_m - h'$, $z_1 = h'_m - h'_1$ e $z_2 = h'_m - h'_2$. O objetivo desta modelagem é que, face ao conhecimento prévio das alturas $h_1$ e $h_2$, a altura $h$ de um objeto qualquer seja determinada.

Usando semelhança de triângulos e equação de reta, obtém-se

$$f = \frac{v_1.z_1}{y_1} = \frac{v_2.z_2}{y_2} = \frac{v.z}{y}, \tag{3.26}$$

onde *f* é a distância focal. Um ponto qualquer no eixo do laser satisfaz a equação da reta, representada pela linha do laser, sendo que os pontos $p_1 = (x_1, y_1, z_1)$ e $p_2 = (x_2, y_2, z_2)$ são conhecidos e pertencentes a esta reta. Logo,

$$\frac{y - y_1}{z - z_1} = \frac{y - y_2}{z - z_2} = \frac{y_2 - y_1}{z_2 - z_1}. \tag{3.27}$$

Substituindo $y, y_1, y_2$ das equações (3.26) em função das equações (3.27), obtém-se que

$$\frac{v.z - v_1.z_1}{z - z_1} = \frac{v.z - v_2.z_2}{z - z_2} = \frac{v_2.z_2 - v_1.z_1}{z_2 - z_1}. \tag{3.28}$$

Tiram-se das equações (3.28) duas constantes: $k = v_2.z_2 - v_1.z_1$ e $d = z_2 - z_1$. Isolando-se a variável *z*, obtém-se

$$z = \frac{z_1.z_2.(v_1 - v_2)}{v.(z_2 - z_1) + v_1.z_1 - v_2.z_2}, \tag{3.29}$$

sendo $n = z_1.z_2.(v_1 - v_2)$ uma constante. Logo, substituindo as 3 constantes na equação (3.29), obtém-se

$$z = \frac{n}{v.d - k}. \tag{3.30}$$

Contudo, devido ao fato de o plano da câmera fazer um ângulo de inclinação $\alpha$ com o plano da superfície, é necessário corrigir as alturas, pois os objetos são medidos perpendicularmente à superfície, sendo $h'_m = h_m / \cos\alpha$, $h'_1 = h_1 / \cos\alpha$, $h'_2 = h_2 / \cos\alpha$ e $h' = h / \cos\alpha$. Logo, a altura *h* de um objeto qualquer é dada por

$$h = h_m - z.\cos\alpha. \tag{3.31}$$

Para a varredura do laser sobre a superfície, foram criados 3 padrões de aquisição de imagens, a saber,

1) ***varredura grossa***: varredura rápida, com 35 linhas de laser para a superfície da imagem;

2) ***varredura padrão***: varredura média, com 70 linhas de laser para a superfície da imagem, sendo este modo de leitura o modo padrão utilizado nesta tese;

3) ***varredura fina***: varredura lenta, com 140 linhas de laser para a superfície. Neste modo de leitura, o tempo de aquisição é maior, e somente é usado caso o sistema queira analisar detalhes bastante nítidos da imagem. Este modo de leitura é usado na calibração do laser.

A Figura 3.14 mostra os objetos da Figura 3.12 no modo de resolução *varredura fina.*

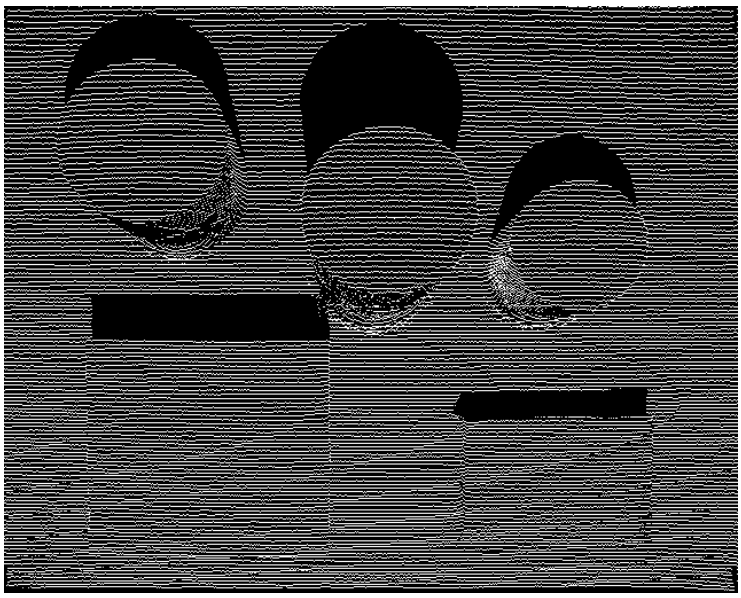

*Figura 3.14: Imagem de uma superfície com 5 objetos em varredura fina.*

Ao aplicar o método proposto para a obtenção da informação de profundidade, aos *pixels* dos perfis de laser da imagem da Figura 3.14, são obtidas as cotas das alturas dos objetos. Estas cotas são associadas a tons de cinza, de forma que objetos cujas alturas variam de 0 a 250 *mm* são associados a *pixels* com valores de 0 a 250 respectivamente.

Contudo, devido ao passo do laser, há regiões entre perfis sem informação de profundidade. Neste caso, é usada a interpolação bilinear para mapear *pixels* nestas regiões. O resultado é a geração da imagem de profundidade em perspectiva.

A Figura 3.15 mostra a imagem de profundidade em perspectiva dos objetos da Figura 3.12.

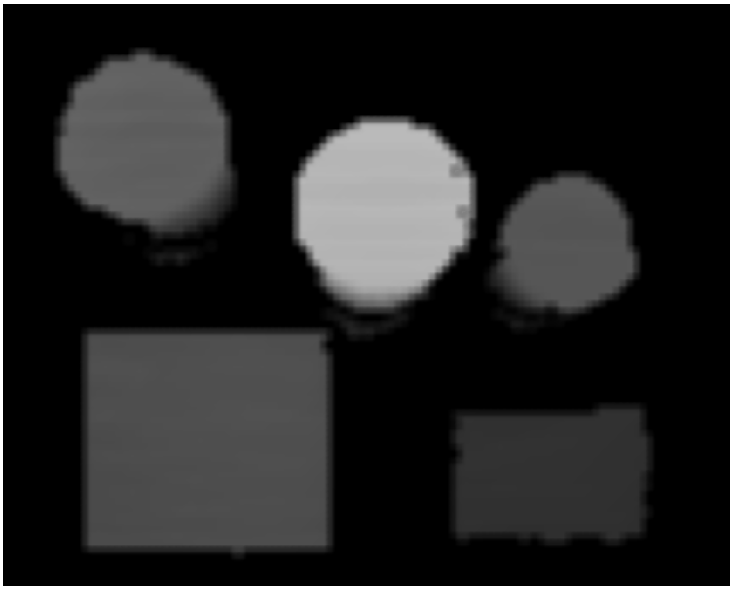

*Figura 3.15: Imagem de profundidade em perspectiva.*

### 3.4.5 Projeção Ortogonal

As imagens adquiridas por câmeras se encontram em perspectiva, com os inúmeros raios de luz advindos de diferentes pontos da imagem passando por um só ponto (centro de perspectiva), localizado no sistema óptico da câmera.

Portanto, devido à perspectiva cônica, objetos mais altos são vistos ocupando áreas maiores, por estarem mais próximos da câmera.

Em uma projeção ortogonal, raios ortogonais são projetados a partir da região da imagem, sendo paralelos, eliminando-se assim a distorção causada pela perspectiva cônica. A Figura 3.16 mostra 3 objetos em perspectiva central e os mesmos 3 objetos em projeção ortogonal.

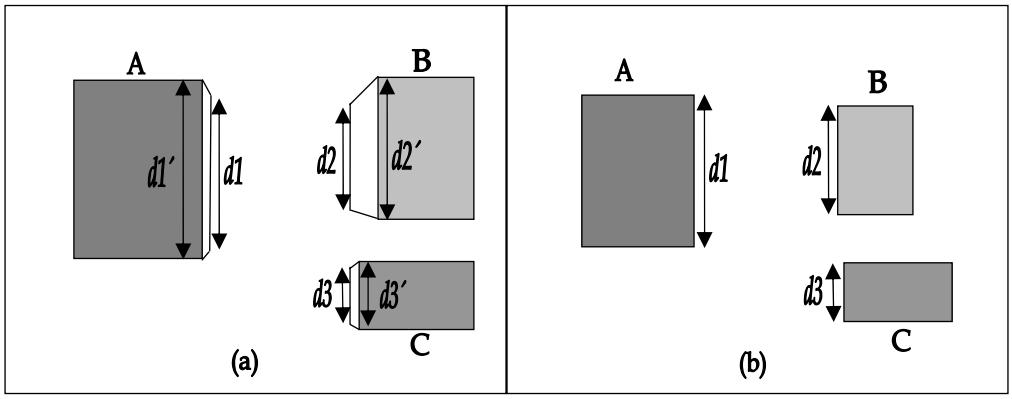


*Figura 3.16: (a) Objetos em perspectiva central (b) Objetos em projeção ortogonal.*

Nota-se que o objeto A é o mais baixo enquanto que o objeto B é o mais alto. Na perspectiva ortogonal, as laterais dos objetos não são mais vistas, devido ao fato de que estas cotas são inferiores às cotas máximas dos objetos, ficando assim ocultas.

A Figura 3.17 mostra o processo de orto-retificação. Para realizar este processo, é necessário saber as alturas dos objetos (elevação), que no caso de imagens aéreas são obtidas através de DTM (*Digital Terrain Models*), que representam não só as coordenadas de uma imagem como também as altitudes das regiões [43].

Na modelagem proposta, as alturas dos objetos são conhecidas, através das imagens de profundidade em perspectiva. Neste caso, a orto-retificação torna-se um procedimento simplificado e de fácil implementação [43].

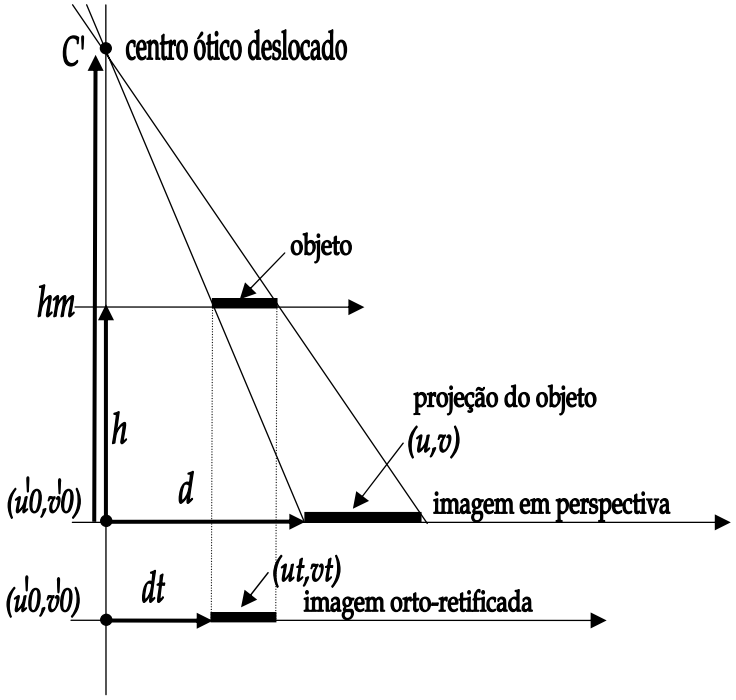


*Figura 3.17: Processo de orto-retificação baseado em semelhança de triângulos.*

Por semelhança de triângulos, sabendo-se que a imagem em perspectiva e a imagem orto-retificada têm as mesmas dimensões, obtém-se

$$u_t = \frac{(u - u'_0).(h_m - h)}{h_m} + u'_0, \tag{3.32}$$

$$v_t = \frac{(v - v'_0).(h_m - h)}{h_m} + v'_0, \tag{3.33}$$

onde $h_m$ é a altura total desde o centro ótico deslocado $C'$ (devido à transformação projetiva) até o plano da superfície, $h$ é a altura de um objeto qualquer, $d$ é a distância radial do ponto $(u,v)$ até $(u'_0,v'_0)$, e $d_t$ é a distância radial do ponto $(u_t,v_t)$ até $(u'_0,v'_0)$, que é o ponto principal da imagem de profundidade.

Neste processo, a imagem em perspectiva deve ser varrida do centro para a periferia, de forma radial, devido à natureza cônica da perspectiva, conforme ilustrado na Figura 3.18, onde $W$ representa a largura da imagem em *pixels* e $H$ representa a altura da imagem em

*pixels*. Para cada *pixel*, partindo do centro ótico deslocado da imagem em perspectiva, através das equações (3.32) e (3.33) é possível determinar a nova posição deste *pixel* na imagem orto-retificada.

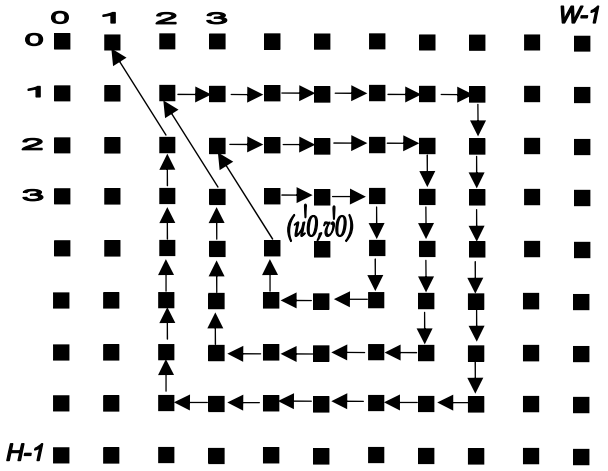


*Figura 3.18: Processo do algoritmo de orto-retificação proposto.*

Como o algoritmo é executado do centro para a periferia, os *pixels* das laterais dos objetos não são mais vistos, porque estes *pixels* têm cotas inferiores aos *pixels* das cotas maiores dos objetos, sendo assim sobrepostos.

A Figura 3.19 mostra a imagem de profundidade orto-retificada, composta pelos 5 objetos de alturas diferentes. Observe-se que os objetos mais altos, aqui representados por tons de cinza próximos ao branco, após a correção, passam a ocupar áreas menores (e reais) aumentando em muito a precisão das medidas dos mesmos.

*Figura 3.19: Imagem de profundidade orto-retificada.*

A Figura 3.20 mostra a imagem de profundidade orto-retificada da Figura 3.19 em vista *3D*.

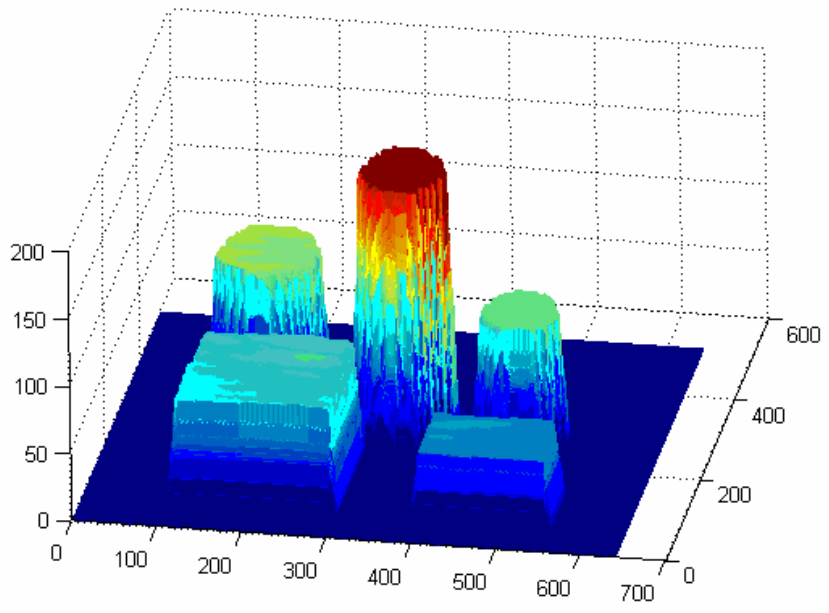


*Figura 3.20: Imagem de profundidade orto-retificada em vista 3D.*

## 3.5 Aquisição e Tratamento de Imagens

A aquisição de imagem é feita através de uma câmera *CCD* com conexão serial padrão *FireWire*, que gera imagens em várias resoluções, com diferentes taxas de quadros por segundo (*frame rate*). A aplicação faz a obtenção da imagem de cada quadro por meio de interrupções, armazenando-as em objetos, para tratamento posterior.

O tratamento da imagem é necessário para adaptar as imagens ao formato propício para extração de informações, além de correções. Algoritmos de tratamento de imagens são usados para este fim. No decorrer desta seção, os principais algoritmos de tratamento de imagens que foram usados neste sistema são explicados.

### 3.5.1 Representação

Uma imagem digital é representada por uma matriz, onde cada ponto (retangular ou quadrado) é um *pixel* (*picture element*), a cujas coordenadas $(x, y)$ corresponde um valor que representa a sua intensidade, dado por $f(x, y)$. A Figura 3.21 mostra a representação gráfica de uma imagem [1].

Devido a detalhes construtivos dos monitores de vídeo, o sistema de coordenadas da imagem é discretizado, não aceitando coordenadas negativas, além de o eixo vertical crescer no sentido para baixo. *W* representa a largura em *pixels* enquanto que *H* representa a altura em *pixels* da imagem, com dimensão *WxH pixels*.

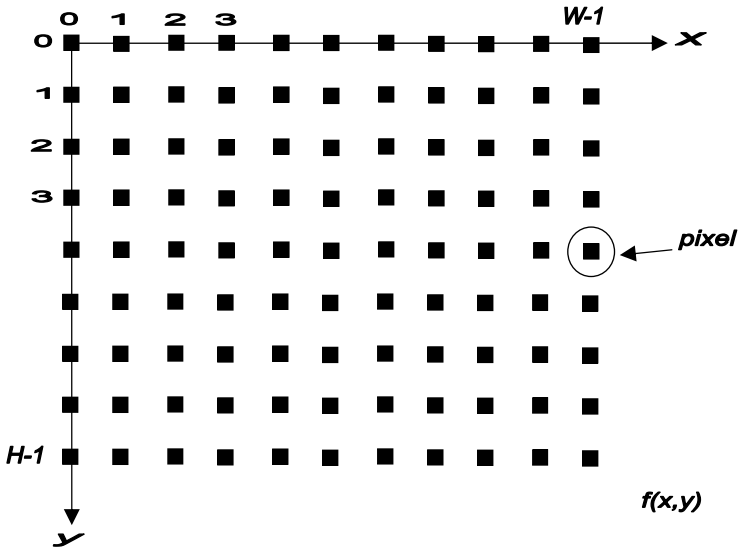


*Figura 3.21: Representação da imagem.*

Usando a notação matricial, é possível escrever a imagem digital, de dimensão *WxH*, como

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & f(0,3) & \cdots & f(0,W-1) \\ f(1,0) & f(1,1) & f(1,2) & f(1,3) & \cdots & f(1,W-1) \\ f(2,0) & f(2,1) & f(2,2) & f(2,3) & \cdots & f(2,W-1) \\ f(3,0) & f(3,1) & f(3,2) & f(3,3) & \cdots & f(3,W-1) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f(H-1,0) & f(H-1,1) & f(H-1,2) & f(H-1,3) & \cdots & f(H-1,W-1) \end{bmatrix}. \quad (3.34)$$

Por simplicidade de notação, a matriz (3.34) pode ser escrita de forma mais tradicional como [1]

$$\mathbf{A}=\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} & \cdots & a_{0,W-1} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,W-1} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,W-1} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,W-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{H-1,0} & a_{H-1,1} & a_{H-1,2} & a_{H-1,3} & \cdots & a_{H-1,W-1} \end{bmatrix}, \quad (3.35)$$

onde $a_{i,j}=f(i,j)$.

### 3.5.2 Aquisição

A aquisição das imagens é feita por interrupção, de modo que o sistema de controle do laser posiciona o laser e envia um sinal de sincronismo para que seja obtida a imagem. Assim, a cada nova posição do laser haverá uma imagem associada, até que toda a superfície seja varrida.

### 3.5.3 Algoritmos de Tratamento da Imagem

Além dos algoritmos de correção radial, transformação projetiva e transformação ortogonal já discutidos, outros algoritmos foram usados para o tratamento da imagem.

Devido ao tipo de câmera *CCD* que é usada, as imagens obtidas têm formato *RGB,* com 24 *bits* de profundidade, 1280 *pixels* de largura por 1024 *pixels* de altura, sendo estas imagens guardadas em um *buffer* para tratamento e extração de características.

A Figura 3.22 mostra um fluxograma do processo de tratamento das imagens.

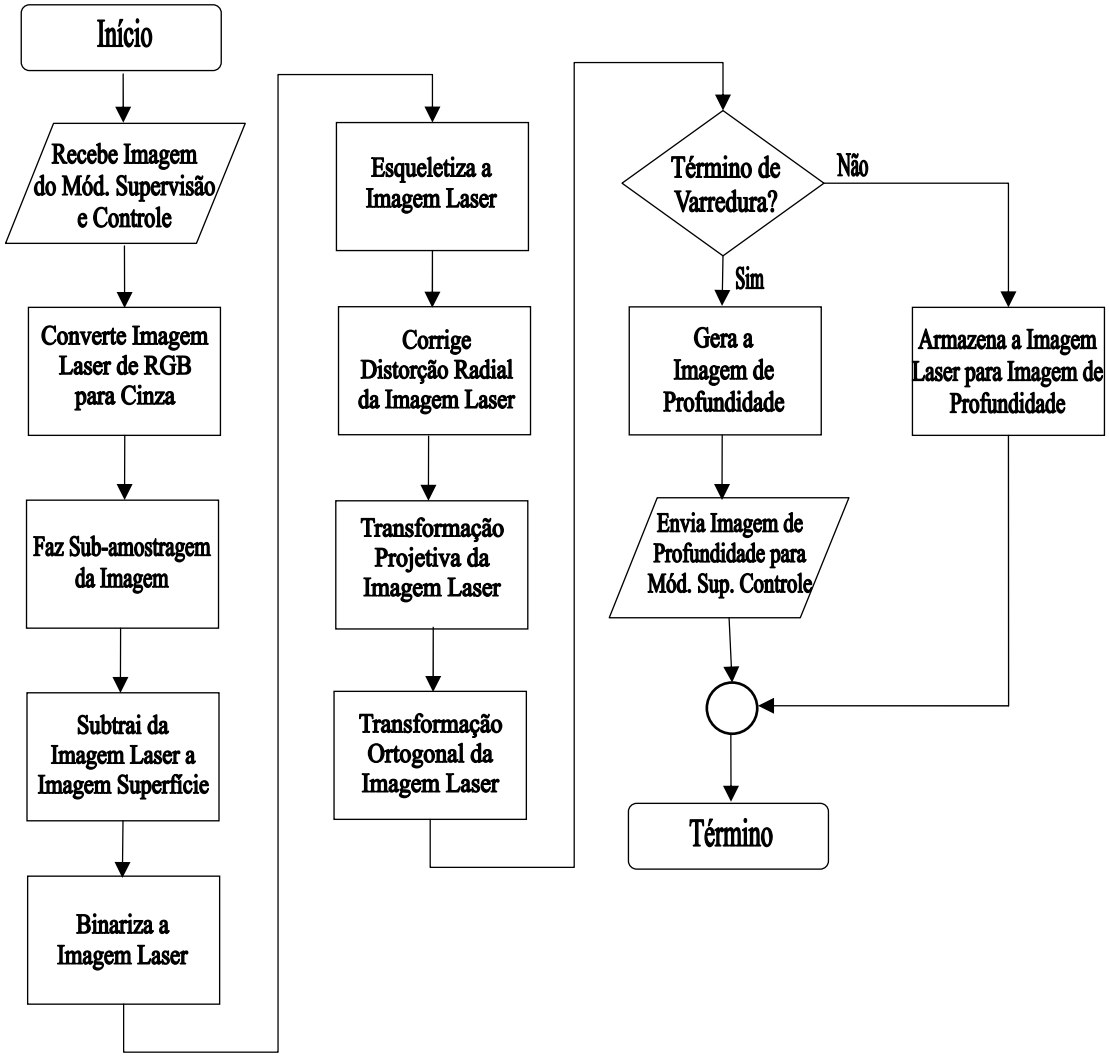


*Figura 3.22: Tratamento da imagem.*

A imagem em estado original é adquirida pelo módulo de supervisão e controle. Inicialmente, ela sofre um processo de conversão para tons de cinza, e é re-amostrada, para redução do seu tamanho original.

Para extração do perfil do laser somente, esta imagem é subtraída da imagem de fundo da superfície, se for o primeiro quadro. Para os quadros seguintes, ela é subtraída do quadro anterior, para otimização e garantia de que fatores externos, como a mudança de luz ambiente entre os quadros, não interfiram.

Posteriormente, a imagem é binarizada para aumentar o contraste do perfil do laser. Então ela é esqueletizada, tornando o perfil uma linha fina e representativa dos contornos dos obstáculos.

Neste estágio, a imagem passa por 3 correções geométricas: correção da distorção radial da lente, transformação projetiva e transformação ortogonal (orto-retificação).

Finalmente, todas as imagens da varredura são somadas e, através do modelo matemático proposto, são obtidas as informações de profundidade dos objetos. Uma interpolação bilinear é feita para preencher toda a imagem, gerando assim a imagem de profundidade.

A seguir, os algoritmos do tratamento de imagem são descritos.

#### 3.5.3.1 Conversão de RGB para Tons de Cinza

Devido à aquisição de imagens ser no formato *RGB*, padrão que exige grande processamento, é necessário converter as imagens para o padrão de *Tons de Cinza*. No padrão *RGB*, para cada *pixel* há 3 *bytes* de intensidade, sendo 1 *byte* para o canal vermelho R (*red*), 1 *byte* para o canal verde G (*green*) e 1 *byte* para o canal azul B (*blue*). Já no padrão *Tons de Cinza*, para cada *pixel* há apenas 1 *byte*, variando de 0 (preto) a 255 (branco).

Esta conversão foi otimizada para que a componente R do formato *RGB* tivesse um peso maior, funcionando também como um filtro digital, de modo que o sinal do laser, que é vermelho, fosse realçado na conversão. A conversão foi feita tomando-se

$$f(x,y) = 0{,}8.R(x,y) + 0{,}1.G(x,y) + 0{,}1.B(x,y), \tag{3.36}$$

onde $f(x,y)$ é o valor da intensidade do tom de cinza, $R(x,y)$ é o valor da componente vermelha, $G(x,y)$ é o valor da componente verde e $B(x,y)$ é o valor da componente azul.

### 3.5.3.2 Sub-amostragem da Imagem

A sub-amostragem da imagem é necessária para reduzir o tempo computacional dos algoritmos. A imagem original adquirida tem largura de 1280 *pixels* por altura de 1024 *pixels*. A imagem sub-amostrada armazena apenas uma linha e uma coluna para cada duas linhas e duas colunas da imagem original, reduzindo a largura e altura da imagem à metade, definindo assim uma imagem em tons de cinza de 640 x 512 pixels.

### 3.5.3.3 Remoção da Imagem de Fundo

A remoção da imagem de fundo (*background*) é necessária devido ao fato de que o sistema não dispõe de um filtro ótico passa-banda para deixar passar somente o espectro do laser para a câmera *CCD*.

Subtraindo-se da imagem total, que contém o laser e a superfície, a imagem da superfície sem laser, obtém-se somente o contorno do laser na superfície. Em outras palavras, toma-se

$$f_L(x,y) = f_{LS}(x,y) - f_S(x,y), \tag{3.37}$$

onde $f_L(x,y)$ é a imagem que contém somente o contorno do laser, $f_{LS}(x,y)$ contém laser e superfície e $f_S(x,y)$ contém a superfície sem a incidência do laser. O efeito final é uma imagem com a remoção do fundo. Para otimizar o processo, a imagem $f_S(x,y)$ usada é a do passo anterior do laser, minimizando assim qualquer variação de luz ambiente entre os quadros sucessivos de aquisição do laser.

### 3.5.3.4 Binarização

Imagens binárias são bastante compactas, sendo muito úteis em sistemas que envolvem grande quantidade de processamento. A binarização é um processo de conversão de uma imagem de Tons de Cinza em uma imagem Monocromática. Assim, apenas 1 *bit* é associado a cada *pixel*, cujo valor zero indica cor preta e valor 1 indica cor branca.

Pelo fato de o padrão adotado para o projeto ser Tons de Cinza, a binarização tem como objetivo permitir extrair da imagem o local da incidência do laser sobre os objetos da superfície. Logo, para uma imagem de origem em tons de cinza $f_0(x,y)$, a imagem de destino binarizada é expressa por

$$f_d(x,y) = \begin{cases} 0, & f_o(x,y) < th \\ 255, & f_o(x,y) \geq th \end{cases}, \tag{3.38}$$

onde o valor *th* (*threshold*) define o limiar de conversão, sendo um valor inteiro que varia de acordo com a tonalidade de cinza da imagem de origem e a região de interesse a ser explorada. O valor *th* foi definido experimentalmente, sendo constante para todo o processo de tratamento da imagem. A função $f_d(x,y) = 0$ representa *pixels* em preto enquanto que $f_d(x,y) = 255$ representa *pixels* em branco.

#### 3.5.3.5 Esqueletização

O processo de esqueletização é implementado em termos das operações de erosão (*erosion*) e abertura (*opening*) aplicadas à imagem [1], sendo iterativo e de grande custo computacional. Portanto, para esta aplicação, um algoritmo de esqueletização simplificado foi desenvolvido, com o objetivo de tornar este processo mais rápido e compatível com os tempos de processamento do sistema.

Basicamente, o perfil de laser obtido após a binarização é de uma largura variável, em função de como ocorre a incidência do laser sobre os objetos. Objetos mais claros refletem melhor a luz, gerando um perfil mais definido e grosso, enquanto objetos mais escuros atenuam mais o laser, e o perfil é mais fino e mais difícil de detectar.

Desta forma, para uniformizar o tratamento dos *pixels* dos perfis de laser, para cada coluna da imagem o algoritmo verifica a quantidade de *pixels* que representa o perfil e escolhe apenas um, no ponto médio, para ser o *pixel* representativo daquela coluna. O processo é repetido para todas as colunas. Este algoritmo encontra o pixel médio $(x,\overline{y})$ apenas nas

colunas, portanto ele faz a varredura (passada) vertical somente, reduzindo os tempos de execução. O valor de $\overline{y}$ é dado por

$$\overline{y} = \frac{y_m + y_n}{2}, \tag{3.39}$$

onde $y_m$ é o número da linha do primeiro *pixel* do perfil e $y_n$ o do último *pixel*. A Figura 3.23 mostra o processo. Os *pixels* em branco indicam o perfil da linha de laser sobre a superfície da imagem após a binarização. Com a esqueletização, apenas um *pixel* médio por coluna é usado, reduzindo assim o tempo computacional do processo de geração da imagem de profundidade.

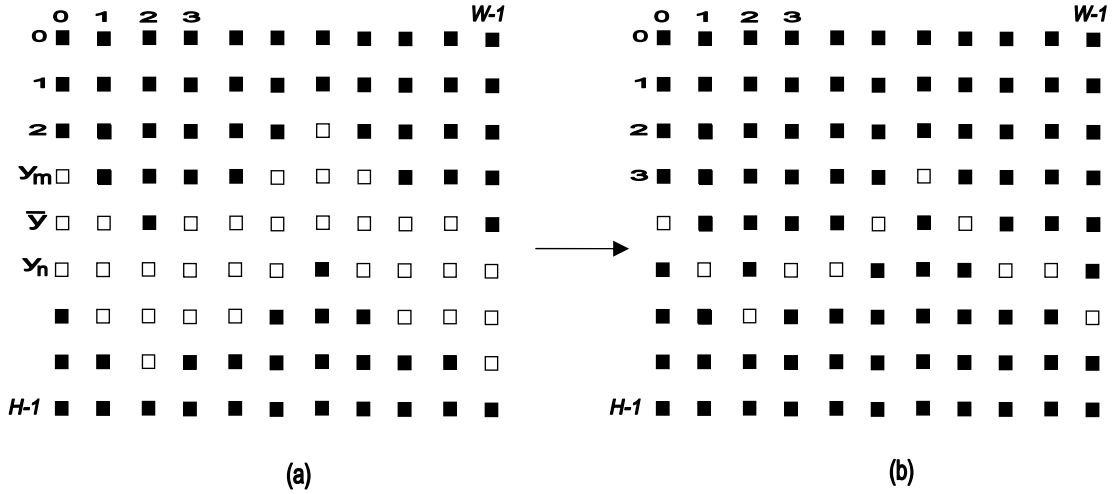


*Figura 3.23: (a) Perfil binarizado da linha de laser sobre a superfície e (b) Linha de laser esqueletizada.*

### 3.5.3.6 Imagens de Profundidade

As imagens de profundidade (*range images*), também conhecidas como mapas de profundidade ou mapas *XYZ*, permitem reproduzir a estrutura *3D* de uma superfície. Elas já codificam a posição da superfície diretamente, pois cada *pixel* expressa a distância de uma referência conhecida da imagem em relação a um ponto visível da superfície.

As imagens de profundidade são representadas por uma matriz de valores de profundidade, e podem ser associadas a imagens em tons de cinza, em que a intensidade $f(x, y)$ de um

determinado *pixel* representa, de forma quantificada, a profundidade de um determinado ponto, em relação a um referencial.

As linhas de laser, uma vez incidindo sobre uma superfície com objetos, permitem capturar os seus contornos, e estas cotas são obtidas pela modelagem matemática proposta. Cada linha de laser gera um segmento da imagem de profundidade, e posteriormente todos os segmentos são adicionados em uma única imagem. A obtenção das profundidades onde se encontra o laser é feita após um processo de calibração do sistema, baseada na modelagem matemática descrita nos itens 3.4.3, 3.4.4 e 3.4.5.

# Capítulo 4 - Mapeamento da Superfície de Navegação do Robô Guará

## 4.1 Introdução

Neste capítulo são apresentados os modelos utilizados para o mapeamento da superfície de navegação e dos seus obstáculos. Inicialmente, é feita a definição dos sistemas de coordenadas, que tem como objetivo criar modelos matemáticos capazes de mapear os pontos *3D* do espaço dos obstáculos em relação ao centro ótico da câmera *CCD* e do referencial do quadro, e posteriormente a transformação geométrica (rotação e translação) destes pontos em relação ao sistema de coordenadas do robô Guará e do mundo.

Do sistema de visão são obtidas imagens de profundidade precisas, que indicam a existência ou não de obstáculos na superfície. Estas imagens, no geral, não mostram os obstáculos totalmente, devido à limitação do campo de visão da câmera *CCD* e também do alcance do laser. A Figura 4.1 mostra esta limitação.

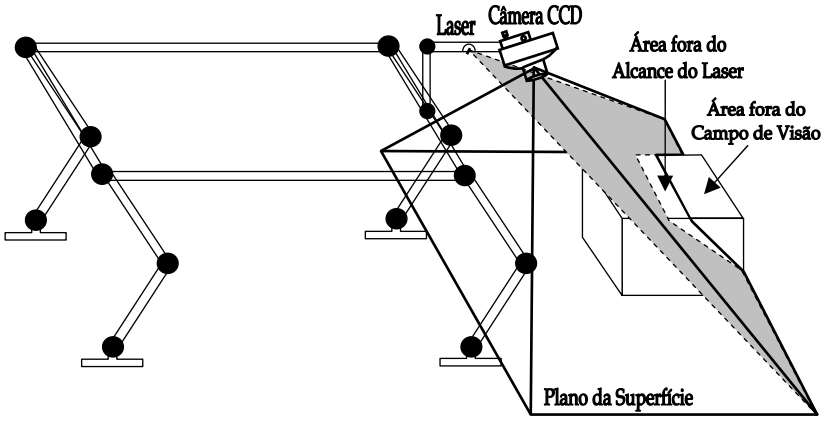


*Figura 4.1: Campo de visão da câmera CCD e alcance do laser.*

É necessário que o robô Guará dê novos passos para que os obstáculos sejam totalmente mapeados. Assim, as imagens de profundidade de quadros sucessivos devem ser integradas,

para o mapeamento dos obstáculos. O processo de integração de imagens é conhecido como *registro de imagens*.

Durante o registro de imagens, áreas comuns entre imagens sucessivas geram pontos redundantes que são descartados, escolhendo-se apenas pontos representativos para estas imagens integradas. Estes pontos estão eqüidistantes e geram obstáculos elementares, com áreas e volumes conhecidos através de uma técnica de *cubificação*.

O registro de imagens é baseado em dados de odometria do robô Guará. Contudo, estes dados têm incertezas associadas à leitura de posição dos potenciômetros das juntas e ao escorregamento das patas. Para aumentar a precisão do mapeamento, foi criado um algoritmo iterativo do ponto mais próximo (*ICP*): caso haja áreas comuns em imagens sucessivas, o *ICP* gera uma matriz de transformação para corrigir o alinhamento entre as imagens, e, assim, o sistema de odometria do robô é atualizado. Sua implementação é descrita no decorrer deste capítulo.

O obstáculo é representado por uma quantidade de obstáculos elementares vizinhos. Caso o robô retorne ao mesmo local pela segunda vez, os pontos redundantes da nova imagem são descartados, devido ao mapeamento prévio daquele local, e o obstáculo é atualizado, com pontos extras representativos incluídos pela nova imagem de profundidade. O obstáculo é, então, re-cubificado.

A partir daí uma série de descritores de obstáculos, com baixo custo computacional de processamento, são gerados para auxiliar o robô durante a sua navegação.

## 4.2 Referenciais

Para se mudar de um referencial para outro, uma seqüência de operações matriciais de rotações e translações sobre pontos *3D* de um referencial devem ser feitas, para que estes pontos sejam transformados em pontos *3D* de outro referencial.

Os referenciais envolvidos são os seguintes:

- ✓ referenciais do Robô Guará;
- ✓ referenciais do Sistema de Visão;
- ✓ referencial do Mundo.

### 4.2.1 Referenciais do Robô Guará

O modelo cinemático do robô Guará é capaz de fornecer a posição da sua plataforma, a partir das coordenadas de juntas medidas nos potenciômetros das pernas em apoio no solo.

A coordenação dos movimentos do robô é realizada através do modelo cinemático de determinação da orientação da plataforma e da posição do centro geométrico. O robô dispõe de 4 pernas com 4 graus de liberdade cada uma, que funcionam como manipuladores, fixados na plataforma do robô e cujos espaços operacionais são definidos pelas trajetórias das extremidades das patas.

O movimento do robô pressupõe o acionamento das patas segundo uma trajetória determinada no espaço cartesiano, mapeada no espaço das articulações pela cinemática inversa da perna [2]. A partir das informações de coordenadas dos pontos de apoio das patas no espaço cartesiano, fornecidas pela cinemática direta da perna, é obtida a orientação da plataforma e a posição do centro geométrico do robô.

Para efeito da análise cinemática, a plataforma e as pernas do robô são consideradas um sistema de corpos rígidos. Os pontos de apoio das patas formam sempre um plano que será paralelo à superfície de apoio.

Na partida, o robô posiciona a plataforma com uma elevação *e* do solo, comandando ângulos iguais para juntas equivalentes de cada perna, de acordo com a sua cinemática inversa. A Figura 4.2 mostra os referenciais do robô Guará.

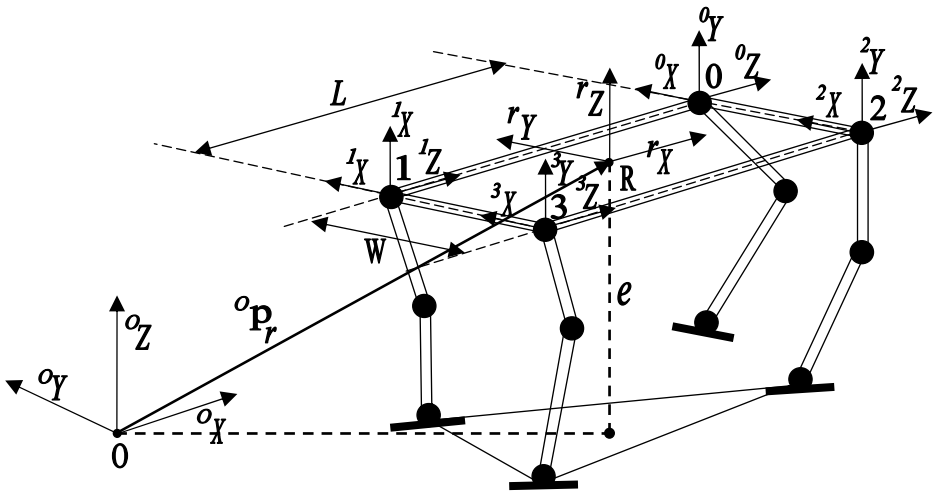


*Figura 4.2: Referenciais do robô Guará.*

*L* e *W* são comprimento e largura, respectivamente, entre pontos de fixação das pernas; *O* é o referencial fixo de Origem; *R* é o referencial do Robô instalado a bordo e fixo no centro geométrico da plataforma; *0* é o *referencial da perna 0*, *1* é o *referencial da perna 1, 2* é o *referencial da perna 2* e *3* é o *referencial da perna 3,* fixos no ponto de conexão da perna ao robô. As pernas são numeradas da seguinte forma: *dianteira esquerda*, número *0*; *traseira esquerda*, número *1*; *dianteira direita*, número *2*, e *traseira direita*, número *3* [2].

Cada perna do robô Guará é formada por um manipulador de quatro graus de liberdade preso a uma plataforma que se locomove. A Figura 4.3 mostra uma vista lateral do robô, indicando as 4 pernas, as juntas e referenciais das pernas *2* e *3* e o do robô *R*.

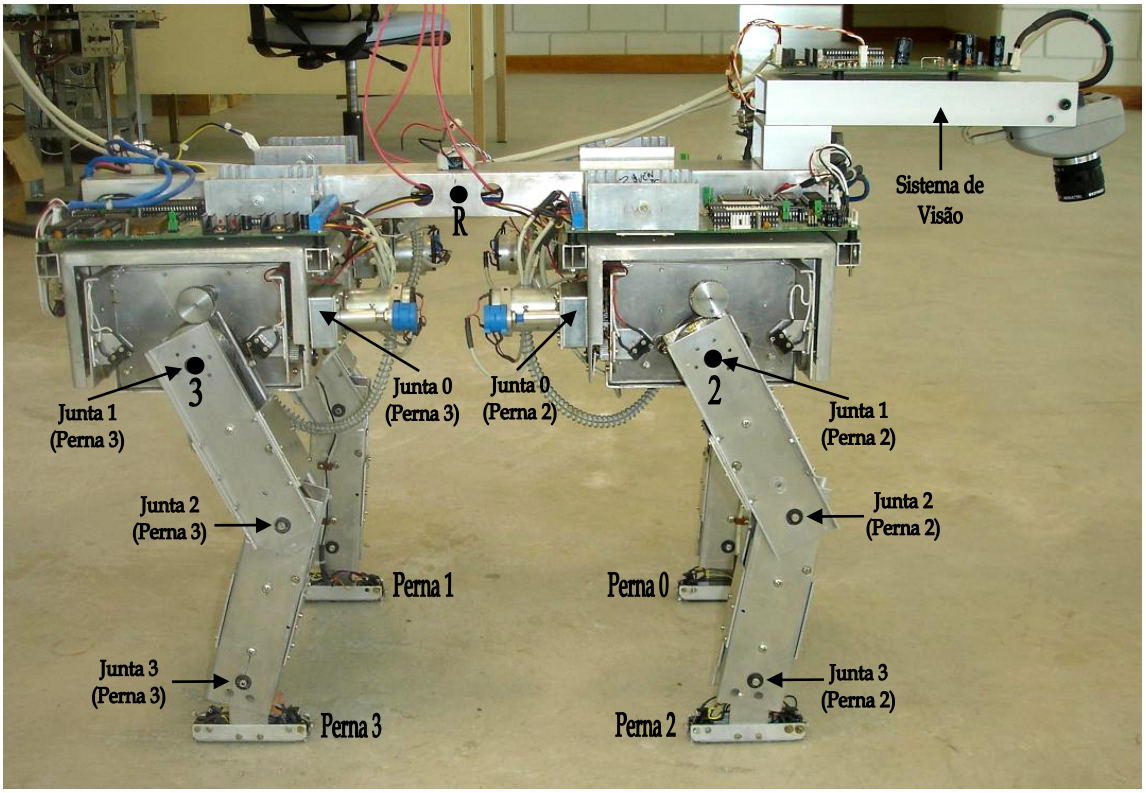


*Figura 4.3: Vista lateral do robô indicando as 4 pernas e juntas.*

A Figura 4.4 mostra o sistema de coordenadas de uma perna com os 4 graus de liberdade, de acordo com a convenção de *Denavit-Hartenberg.* A perna foi construída de maneira que os graus de liberdade *0* e *1* tenham a mesma origem, com o comprimento do primeiro segmento nulo. Os graus de liberdade são mergulho na junta 0 e rolagem na junta 1. As juntas 2 e 3 só têm mergulho [2].

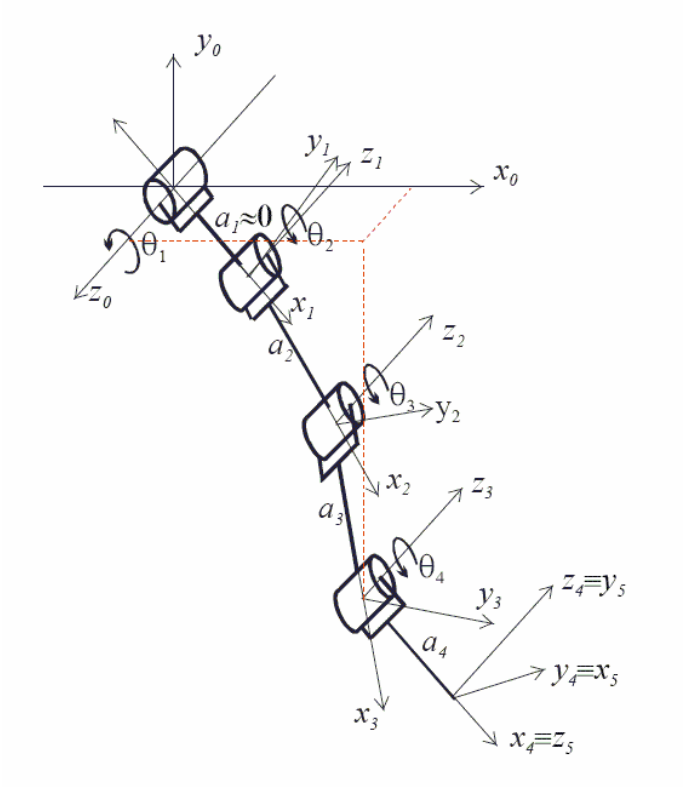


*Figura 4.4: Sistemas de coordenadas de uma perna do Guará.*

De acordo com a Figura 4.2, em relação ao referencial do robô, as coordenadas dos pontos de fixação das pernas *0, 1, 2* e *3* são, respectivamente,

$$\begin{aligned}
\left({}^{r}_{0}x_0,\ {}^{r}_{0}y_0,\ {}^{r}_{0}z_0\right) &= \left(\frac{L}{2},\ \frac{W}{2},\ 0\right),\\
\left({}^{r}_{0}x_1,\ {}^{r}_{0}y_1,\ {}^{r}_{0}z_1\right) &= \left(-\frac{L}{2},\ \frac{W}{2},\ 0\right),\\
\left({}^{r}_{0}x_2,\ {}^{r}_{0}y_2,\ {}^{r}_{0}z_2\right) &= \left(\frac{L}{2},-\frac{W}{2},\ 0\right),\\
\left({}^{r}_{0}x_3,\ {}^{r}_{0}y_3,\ {}^{r}_{0}z_3\right) &= \left(-\frac{L}{2},\ -\frac{W}{2},\ 0\right),
\end{aligned} \tag{4.1}$$

onde o índice superior esquerdo indica o referencial, o índice inferior esquerdo indica o número da junta e o índice inferior direito identifica a pata.

Em relação às coordenadas das pernas, as coordenadas dos pontos de apoio das patas na superfície em relação à plataforma são definidas por

$$
\begin{aligned}
{}^{0}p_0 &= \left( {}^{0}_{4}x_0,\ {}^{0}_{4}y_0, {}^{0}_{4}z_0 \right), \\
{}^{1}p_1 &= \left( {}^{1}_{4}x_1,\ {}^{1}_{4}y_1,\ {}^{1}_{4}z_1 \right), \\
{}^{2}p_2 &= \left( {}^{2}_{4}x_2,\ {}^{2}_{4}y_2,\ {}^{2}_{4}z_2 \right), \\
{}^{3}p_3 &= \left( {}^{3}_{4}x_3,\ {}^{3}_{4}y_3,\ {}^{3}_{4}z_3 \right),
\end{aligned}
\tag{4.2}
$$

onde o índice superior esquerdo de *p* indica o referencial e o índice inferior direito indica a identificação da pata. Quanto às coordenadas, o índice superior esquerdo indica o referencial, o índice inferior esquerdo indica a junta 4 (de apoio) e o índice inferior direito indica a pata.

Em relação ao referencial do robô, os vetores

$$
\begin{aligned}
{}^{r}\mathbf{p}_0 &= \begin{bmatrix} {}^{r}_{4}x_0 + \dfrac{L}{2} & {}^{r}_{4}y_0 + \dfrac{W}{2} & {}^{r}_{4}z_0 \end{bmatrix}^T, \\
{}^{r}\mathbf{p}_1 &= \begin{bmatrix} {}^{r}_{4}x_1 - \dfrac{L}{2} & {}^{r}_{4}y_1 + \dfrac{W}{2} & {}^{r}_{4}z_1 \end{bmatrix}^T, \\
{}^{r}\mathbf{p}_2 &= \begin{bmatrix} {}^{r}_{4}x_2 + \dfrac{L}{2} & {}^{r}_{4}y_2 - \dfrac{W}{2} & {}^{r}_{4}z_2 \end{bmatrix}^T, \\
{}^{r}\mathbf{p}_3 &= \begin{bmatrix} {}^{r}_{4}x_3 - \dfrac{L}{2} & {}^{r}_{4}y_3 - \dfrac{W}{2} & {}^{r}_{4}z_3 \end{bmatrix}^T,
\end{aligned}
\tag{4.3}
$$

são determinados, onde *W* é a distância entre os pontos de fixação de 2 patas em lados distintos, e *L* é a distância entre os pontos de fixação de 2 patas em um mesmo lado, respectivamente, e

$$
\begin{bmatrix} {}^{r}_{4}x_i \\ {}^{r}_{4}y_i \\ {}^{r}_{4}z_i \end{bmatrix} = {}^{ir}\mathbf{R}.\begin{bmatrix} {}^{i}_{4}x_i \\ {}^{i}_{4}y_i \\ {}^{i}_{4}z_i \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.\begin{bmatrix} {}^{i}_{4}x_i \\ {}^{i}_{4}y_i \\ {}^{i}_{4}z_i \end{bmatrix},
\tag{4.4}
$$

onde os índices superiores esquerdos indicam o referencial, os índices inferiores esquerdos indicam o número da junta e os índices inferiores direitos indicam a pata, e ${}^{ir}\mathbf{R}$ é a matriz constante que dá a rotação do referencial da perna em relação ao referencial do robô, sendo que suas colunas são as coordenadas do referencial do robô em relação ao referencial da perna.

Existem quatro combinações possíveis de três patas no solo: *012, 013, 023* e *123*, mostradas na Figura 4.5 a seguir, em vista superior. Apesar de dispostos de forma retangular, os pontos de apoio podem ocupar qualquer coordenada no espaço de alcance das pernas no plano de apoio.

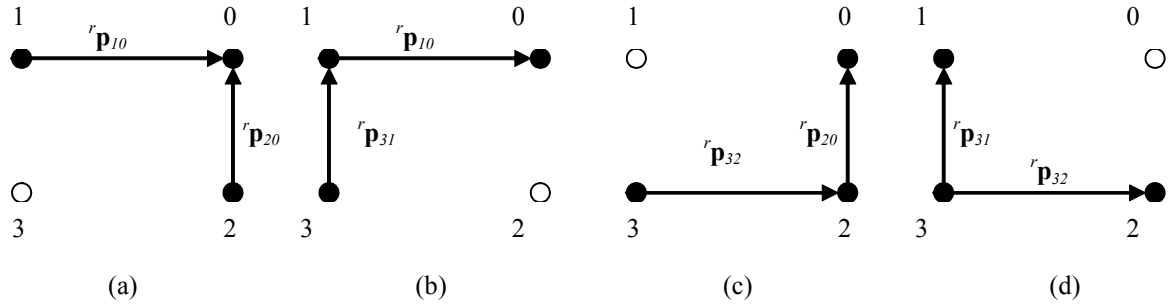


*Figura 4.5: Configurações para 3 patas no solo: (a) patas 0, 1 e 2; (b) patas 0, 1 e 3; (c) patas 0, 2 e 3 e (d) patas 1, 2 e 3. Fonte: [2].*

Definem-se os vetores de posição de um ponto de apoio em relação a outro, orientados de acordo com o referencial do robô, de acordo com as configurações mostradas na Figura 4.5, como

$$\begin{aligned} {}^r\mathbf{p}_{10} &= {}^r\mathbf{p}_0 - {}^r\mathbf{p}_1, \\ {}^r\mathbf{p}_{20} &= {}^r\mathbf{p}_0 - {}^r\mathbf{p}_2, \\ {}^r\mathbf{p}_{31} &= {}^r\mathbf{p}_1 - {}^r\mathbf{p}_3, \\ {}^r\mathbf{p}_{32} &= {}^r\mathbf{p}_2 - {}^r\mathbf{p}_3. \end{aligned} \tag{4.5}$$

Os vetores de posição unem os pontos de apoio e definem o plano de apoio das patas. Efetuando o produto vetorial dos vetores posição nas configurações da Figura 4.5, na ordem dos vetores paralelos para os perpendiculares ao eixo longitudinal da plataforma, de acordo com a regra da mão direita, obtém-se o vetor normal ao plano de apoio, em cada configuração de apoio, da forma

$$\begin{aligned} {}^r_{012}\mathbf{p} &= {}^r\mathbf{p}_{10} \times {}^r\mathbf{p}_{20} = S\left({}^r\mathbf{p}_{10}\right).{}^r\mathbf{p}_{20} \ , \\ {}^r_{013}\mathbf{p} &= {}^r\mathbf{p}_{10} \times {}^r\mathbf{p}_{31} = S\left({}^r\mathbf{p}_{10}\right).{}^r\mathbf{p}_{31} \ , \\ {}^r_{023}\mathbf{p} &= {}^r\mathbf{p}_{32} \times {}^r\mathbf{p}_{20} = S\left({}^r\mathbf{p}_{32}\right).{}^r\mathbf{p}_{20} \ , \\ {}^r_{123}\mathbf{p} &= {}^r\mathbf{p}_{32} \times {}^r\mathbf{p}_{31} = S\left({}^r\mathbf{p}_{32}\right).{}^r\mathbf{p}_{31} \ , \end{aligned} \tag{4.6}$$

onde

$$S\left({}^{r}\mathbf{p}_{ij}\right) = \begin{bmatrix} 0 & -{}^{r}p_{ij}^{x} & {}^{r}p_{ij}^{y} \\ {}^{r}p_{ij}^{x} & 0 & -{}^{r}p_{ij}^{z} \\ -{}^{r}p_{ij}^{y} & {}^{r}p_{ij}^{z} & 0 \end{bmatrix} . \tag{4.7}$$

Os vetores unitários são, então,

$${}^{r}_{mnp}\mathbf{n} = \frac{{}^{r}_{mnp}\mathbf{p}}{\left\| {}^{r}_{mnp}\mathbf{p} \right\|} = \begin{bmatrix} {}^{r}_{mnp}n^{x} & {}^{r}_{mnp}n^{y} & {}^{r}_{mnp}n^{z} \end{bmatrix}^{T}, \tag{4.8}$$

onde *mnp = 012, 013, 023* ou *123*, ${}^{r}_{mnp}\mathbf{n}$ é o vetor unitário na direção normal ao plano de apoio, e ${}^{r}_{mnp}n^{x}$, ${}^{r}_{mnp}n^{y}$ e ${}^{r}_{mnp}n^{z}$ são as suas componente *x*, y e z.

Definem-se os vetores unitários na direção ${}^{r}_{mnp}\mathbf{x}$ a partir dos vetores posição que unem os pontos de apoio de duas patas de um mesmo lado do robô em cada configuração, como

$$\begin{aligned} {}^{r}_{012}\mathbf{x} &= \frac{{}^{r}\mathbf{p}_{10}}{\left\| {}^{r}\mathbf{p}_{10} \right\|} = \begin{bmatrix} {}^{r}_{012}x^{x} & {}^{r}_{012}x^{y} & {}^{r}_{012}x^{z} \end{bmatrix}^{T}, \\ {}^{r}_{013}\mathbf{x} &= {}^{r}_{012}\mathbf{x}, \\ {}^{r}_{023}\mathbf{x} &= \frac{{}^{r}\mathbf{p}_{32}}{\left\| {}^{r}\mathbf{p}_{32} \right\|} = \begin{bmatrix} {}^{r}_{023}x^{x} & {}^{r}_{023}x^{y} & {}^{r}_{023}x^{z} \end{bmatrix}^{T}, \\ {}^{r}_{123}\mathbf{x} &= {}^{r}_{023}\mathbf{x}. \end{aligned} \tag{4.9}$$

O vetor unitário ${}^{r}_{mnp}\mathbf{y}$, que completa o referencial, é obtido pelo produto vetorial do vetor unitário normal ${}^{r}_{mnp}\mathbf{n}$ de (4.8) pelo vetor unitário ${}^{r}_{mnp}\mathbf{x}$ na direção *x*, obtido de (4.9), conforme

$${}^{r}_{mnp}\mathbf{y} = {}^{r}_{mnp}\mathbf{n} \times {}^{r}_{mnp}\mathbf{x} = \begin{bmatrix} -{}^{r}_{mnp}n^{x} & {}^{r}_{mnp}x^{y} + {}^{r}_{mnp}n^{y} & {}^{r}_{mnp}x^{z} \\ {}^{r}_{mnp}n^{x} & {}^{r}_{mnp}x^{x} - {}^{r}_{mnp}n^{z} & {}^{r}_{mnp}x^{z} \\ -{}^{r}_{mnp}n^{y} & {}^{r}_{mnp}x^{x} + {}^{r}_{mnp}n^{z} & {}^{r}_{mnp}x^{z} \end{bmatrix} = \begin{bmatrix} {}^{r}_{mnp}y^{x} & {}^{r}_{mnp}y^{y} & {}^{r}_{mnp}y^{z} \end{bmatrix}^{T}, \tag{4.10}$$

$$\begin{aligned} {}^{r}_{023}\mathbf{y} &= {}^{r}_{012}\mathbf{y}, \\ {}^{r}_{123}\mathbf{y} &= {}^{r}_{013}\mathbf{y}. \end{aligned}$$

A matriz de rotação do plano de apoio em relação ao referencial do robô, cujas colunas são as componentes dos vetores unitários nas direções *x*, *y* e *z* do referencial $R$ do robô, é dada por

$$ {}^{r}_{i-1}\mathbf{R}^{i}_{mnp} = \begin{bmatrix} {}^{r}_{mnp}x^{x} & {}^{r}_{mnp}y^{x} & {}^{r}_{mnp}n^{x} \\ {}^{r}_{mnp}x^{y} & {}^{r}_{mnp}y^{y} & {}^{r}_{mnp}n^{y} \\ {}^{r}_{mnp}x^{z} & {}^{r}_{mnp}y^{z} & {}^{r}_{mnp}n^{z} \end{bmatrix}, \tag{4.11} $$

onde $i-1$ e *i* identificam dois acionamentos consecutivos. A matriz de rotação ${}^{r}_{i-1}\mathbf{R}^{i}_{mnp}$ foi obtida a partir das coordenadas dos pontos de apoio das patas em relação ao referencial do robô. Conforme definido, no instante 0, define-se o referencial do mundo $O$, fazendo-se coincidir os eixos ${}^{o}_{0}Z$ do referencial do mundo e ${}^{r}_{0}Z$ do referencial do robô e mantendo-se paralelos, e com mesmo sentido, os eixos ${}^{o}_{0}X \,//\, {}^{r}_{0}X$ e ${}^{o}_{0}Y \,//\, {}^{r}_{0}Y$ do referencial fixo e do referencial do robô, respectivamente.

Então, entre os instantes *0* da partida, com o robô ainda em repouso, e *1*, imediatamente após o primeiro acionamento de movimento das patas, a matriz ${}^{r}_{0}\mathbf{R}^{1}_{mnp}$ representa também a rotação da plataforma em relação ao referencial do mundo ${}^{o}_{0}\mathbf{R}^{1}_{mnp}$.

Após o segundo acionamento, a matriz de rotação obtida será a rotação do referencial do robô em relação a ele próprio na posição anterior, após o primeiro acionamento. Assim, após o acionamento $i-1$ a matriz de rotação obtida será a rotação do referencial do robô em relação a ele próprio, ocorrida entre os acionamentos $i-1$ e *i*.

A matriz de rotação da plataforma em relação ao referencial fixo após o acionamento *i,* será dada por

$$ {}^{o}_{0}\mathbf{R}^{i}_{mnp} = {}^{o}_{0}\mathbf{R}^{0}_{mnp} \cdot \; {}^{r}_{0}\mathbf{R}^{1}_{mnp} \cdot {}^{r}_{1}\mathbf{R}^{2}_{mnp} \cdots {}^{r}_{i-1}\mathbf{R}^{i}_{mnp} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}_{mnp}, \tag{4.12} $$

onde a matriz ${}^{o}_{0}\mathbf{R}^{0}_{mnp}$ é a matriz de rotação da plataforma em relação à horizontal no instante *0*. Esta matriz é uma matriz identidade de ordem *3*, se o robô está apoiado em um plano horizontal no instante *0*.

Em relação ao referencial de origem, o vetor posição do centro geométrico da plataforma no instante zero da partida, com as 4 patas apoiadas no solo é definido por

$$ {}^{o}_{0}\mathbf{p}^{0}_{r} = \begin{bmatrix} 0 & 0 & e \end{bmatrix}^{T}, \tag{4.13} $$

onde o índice superior esquerdo indica o referencial do mundo, o índice inferior direito indica o referencial do robô, o índice inferior esquerdo o instante de análise anterior e o índice superior direito o instante de análise atual, que no caso do instante de partida, são ambos 0.

As patas de apoio permanecem fixas e o referencial do robô se desloca juntamente com a plataforma. Nestas condições, é possível obter a posição atual da plataforma em função da posição atual e anterior das patas [3]. Foi escolhida a pata 2 para a modelagem, mas poderia ser qualquer uma das quatro. A Figura 4.6 mostra a modelagem adotada para determinar o vetor posição do centro geométrico em relação à origem no instante de análise *i*.

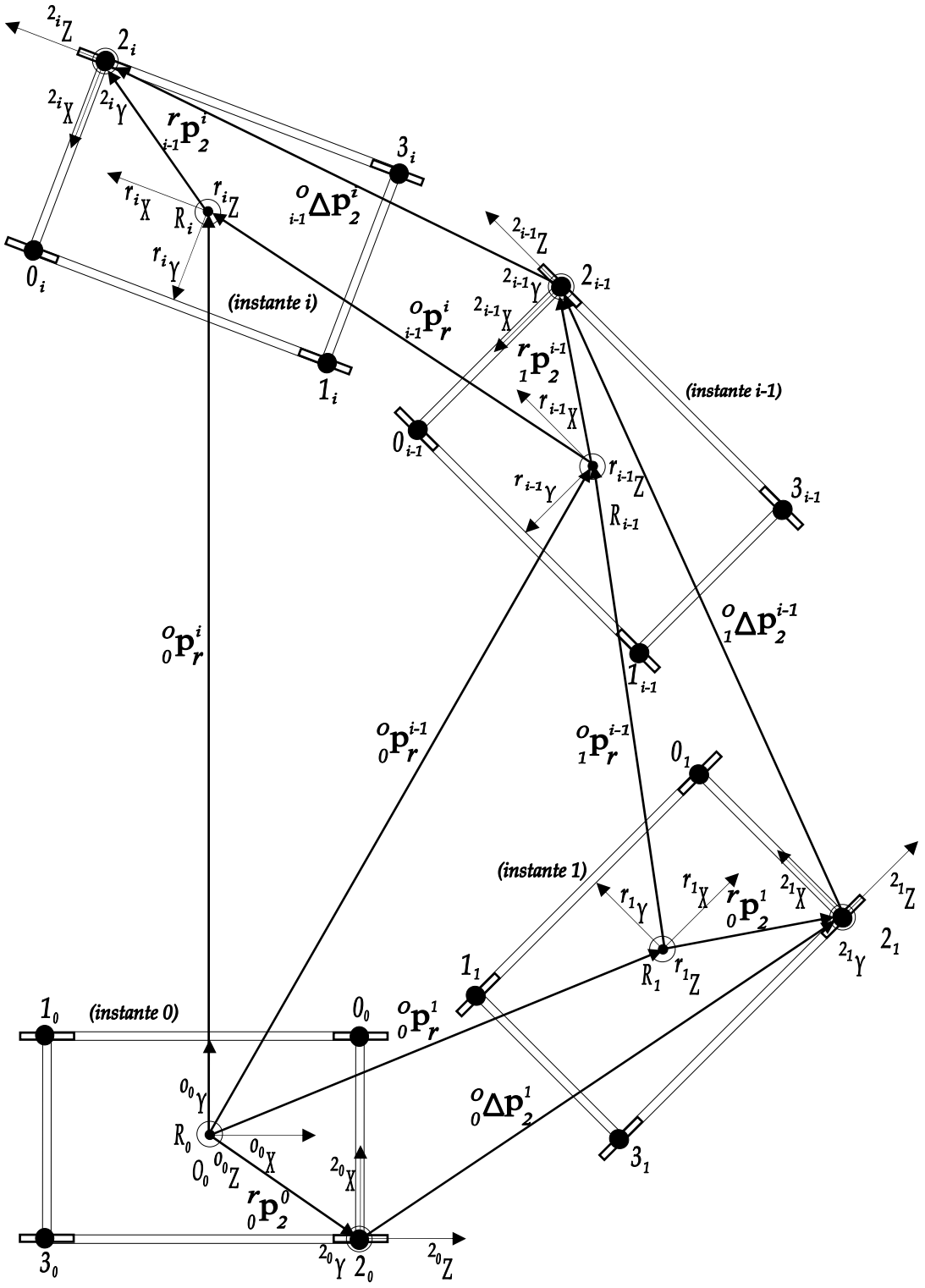


*Figura 4.6: Obtenção do vetor posição do centro geométrico em relação ao referencial do mundo.*

O vetor posição do centro geométrico após o primeiro acionamento é dado por ${}^{o}_{0}\mathbf{p}^{1}_{r}$. Seja ${}^{r}_{0}\mathbf{p}^{0}_{2}$ o vetor posição da origem do referencial da perna 2 em relação a R no instante 0. Seja vetor diferença da origem do referencial da perna 2 entre os instantes 1 e 0, no referencial de origem, dado por ${}^{o}_{0}\Delta\mathbf{p}^{1}_{2} = {}^{o}_{0}\mathbf{p}^{1}_{2} - {}^{o}_{0}\mathbf{p}^{0}_{2} = \left[ {}^{o}_{0}x^{1}_{2} - {}^{o}_{0}x^{0}_{2} \quad {}^{o}_{0}y^{1}_{2} - {}^{o}_{0}y^{0}_{2} \quad {}^{o}_{0}z^{1}_{2} - {}^{o}_{0}z^{0}_{2} \right]$, onde o índice superior da esquerda indica o referencial, o da direita o acionamento e o sub-índice da direita a pata 2.

Nota-se que ${}^{r}_{0}\mathbf{p}^{0}_{2} = {}^{r}_{0}\mathbf{p}^{1}_{2} = {}^{r}_{1}\mathbf{p}^{i-1}_{2} = {}^{r}_{i-1}\mathbf{p}^{i}_{2}$ são valores constantes, pois não há movimento relativo entre o referencial do robô e o referencial da perna 2. O deslocamento do centro geométrico do robô em relação ao referencial do mundo, entre os instantes 0 e 1 é ${}^{o}_{0}\mathbf{p}^{1}_{r}$, dado por

$$ {}^{o}_{0}\mathbf{p}^{1}_{r} = {}^{o}_{0}\mathbf{p}^{0}_{r} + {}^{o}_{0}\mathbf{R}^{1}_{mnp} \cdot {}^{r}_{0}\mathbf{p}^{1}_{2} - {}^{o}_{0}\Delta\mathbf{p}^{1}_{2} + {}^{r}_{0}\mathbf{p}^{0}_{2}. \tag{4.14} $$

Generalizando para os instantes *i-1* e *i*, tem-se que

$$ {}^{o}_{i-1}\Delta\mathbf{p}^{i}_{2} = \begin{bmatrix} {}^{o}_{i-1}x^{i}_{2} - {}^{o}_{i-1}x^{i-1}_{2} \\ {}^{o}_{i-1}y^{i}_{2} - {}^{o}_{i-1}y^{i-1}_{2} \\ {}^{o}_{i-1}z^{i}_{2} - {}^{o}_{i-1}z^{i-1}_{2} \end{bmatrix}, \tag{4.15} $$

e, logo,

$$ {}^{o}_{0}\mathbf{p}^{i}_{r} = {}^{o}_{i-1}\mathbf{p}^{i}_{r} + {}^{o}_{0}\mathbf{R}^{i}_{mnp} \cdot {}^{r}_{i-1}\mathbf{p}^{i}_{2} - {}^{o}_{i-1}\Delta\mathbf{p}^{i}_{2}, \tag{4.16} $$

para $\left[ {}^{o}_{0}(x,y,z)^{i}_{0} \right]^{T} = {}^{o}_{0}\mathbf{R}_{mnp}{}^{i} \cdot \left[ {}^{r}_{0}(x,y,z)^{i}_{2} \right]^{T}$.

A matriz de transformação da plataforma do robô em relação ao referencial do mundo é dada por

$$ {}^{o}_{0}\mathbf{M}^{i}_{r} = \left[\begin{array}{c|c} {}^{o}_{0}\mathbf{R}_{mnp}{}^{i} & {}^{o}_{0}\mathbf{p}^{i}_{r} \\ \hline 0^{t}_{3} & 1 \end{array}\right], \tag{4.17} $$

ou seja,

$$ {}^{o}_{0}\mathbf{M}^{i}_{r} = \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & x_r \\ r_{21} & r_{22} & r_{23} & y_r \\ r_{31} & r_{32} & r_{33} & z_r \\ \hline 0 & 0 & 0 & 1 \end{array}\right]. \tag{4.18} $$

A matriz de transformação ${}^{o}_{0}\mathbf{M}^{i}_{r}$ executa um papel chave no mapeamento, pois juntamente com as matrizes de transformação do sistema de visão garante que os pontos *3D* das imagens de profundidade possam ser referenciados ao referencial do mundo.

A conversão dos pontos *3D* das imagens de profundidade em pontos *3D* no referencial do mundo é o primeiro estágio para o mapeamento dos obstáculos da superfície de navegação do robô Guará.

### 4.2.2 Referenciais do Sistema de Visão

O referencial da câmera é representado pelo seu centro ótico *C*. É necessária uma transformação de referencial que permita que pontos *3D* da superfície mapeada pelo sistema de visão sejam transformados em pontos *3D* no referencial do robô Guará.

A superfície é formada pela integração de imagens de profundidade. O referencial das imagens de profundidade é representado por *Q*, cuja localização é no canto superior esquerdo da imagem (quadro).

O referencial da origem *O* é usado para o mapeamento da superfície. Há matrizes de transformação que mapeiam os pontos do referencial do robô Guará *R* e do referencial do Quadro em relação ao referencial da origem.

Todos os referenciais do sistema são mostrados na Figura 4.7. O centro ótico da câmera está acima da plataforma do robô. O ângulo $\alpha$ representa a inclinação da câmera em relação à superfície. A distância entre o referencial do robô e o da câmera é representada por *d*. As distâncias entre o referencial da câmera e o do quadro são representadas por *a*, *b* e *h*. A cota *e* representa a elevação da plataforma do robô em relação à superfície.

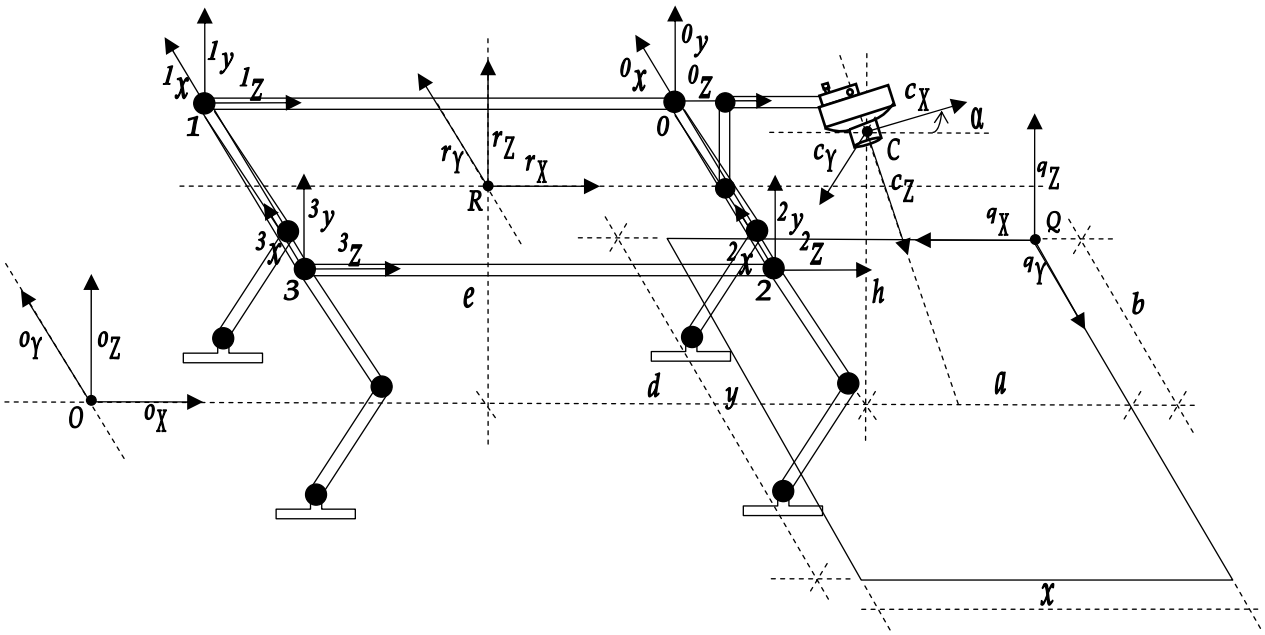


*Figura 4.7: Referenciais da origem, robô, pernas, câmera e quadro.*

## 4.3 Transformações entre Referenciais

Para descrever um sistema de coordenadas *B* em relação ao sistema *A*, deve-se imaginar uma forma de ir de *A* para *B*, ou seja, deve-se determinar uma seqüência de operadores de transformação que transportam o sistema *A*, usado como referência, até que ele coincida com o sistema *B*.

Assim, a transformação homogênea ${}^{a}\mathbf{M}_{b}$ é dita *transformação de A para B*. Ela descreve o sistema *B* em *A*. Logo, a transformação entre dois sistemas de coordenadas pode ser definida por translações e rotações nos eixos que formam tais sistemas.

De acordo com a Figura 4.7, para se passar do referencial *R* (robô) para o referencial *C* (câmera), as seguintes transformações (representadas por matrizes) devem ser executadas, na ordem apresentada:

a) translação *d* no eixo ${}^{r}X$ , dada por

$$ {}^{r}\mathbf{T}_{x,d} = \begin{bmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.19} $$

b) translação (*h-e)* no eixo ${}^{r}Z$ , dada por

$$ {}^{r}\mathbf{T}_{z,(h-e)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & h-e \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.20} $$

c) rotação de 180º no eixo ${}^{r}X$ , dada por

$$ {}^{r}\mathbf{R}_{x,180^o} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(180^o) & -sen(180^o) & 0 \\ 0 & sen(180^o) & cos(180^o) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.21} $$

d) rotação de $\alpha$ graus no eixo ${}^{r}Y$ , dada por

$$ {}^{r}\mathbf{R}_{y,\alpha} = \begin{bmatrix} cos(\alpha) & 0 & -sen(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ sen(\alpha) & 0 & cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.22} $$

Assim, a matriz de transformação do referencial do robô para o referencial da câmera é dada pelo produto das matrizes dos itens *a* até *d*, ou seja,

$$ {}^{r}\mathbf{M}_{c} = {}^{r}\mathbf{T}_{x,d} . {}^{r}\mathbf{T}_{z,(h-e)} . {}^{r}\mathbf{R}_{x,180^o} . {}^{r}\mathbf{R}_{y,\alpha}, $$

$$ {}^{r}\mathbf{M}_{c} = \begin{bmatrix} cos(\alpha) & 0 & -sen(\alpha) & d \\ 0 & -1 & 0 & 0 \\ -sen(\alpha) & 0 & -\cos(\alpha) & h-e \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.23} $$

Dado um vetor no sistema de coordenadas homogêneas do robô ${}^{r}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{r}x & {}^{r}y & {}^{r}z & 1 \end{bmatrix}^T$ e um vetor no sistema de coordenadas homogêneas da câmera ${}^{c}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{c}x & {}^{c}y & {}^{c}z & 1 \end{bmatrix}^T$, tem-se que

$$ {}^{r}\tilde{\mathbf{p}} = {}^{r}\mathbf{M}_{c} . {}^{c}\tilde{\mathbf{p}}. \tag{4.24} $$

A matriz de transformação ${}^{r}\mathbf{M}_{c}$ pode ser expressa em termos de matriz de rotação e vetor de translação, representando assim os *parâmetros extrínsecos* da câmera *CCD* [36], como

$$ {}^{r}\mathbf{M}_{c} = \left[\begin{array}{c|c} {}^{r}\mathbf{R}_{c} & {}^{r}\mathbf{t}_{c} \\ \hline 0_{3}^{t} & 1 \end{array}\right], \tag{4.25} $$

onde ${}^{r}\mathbf{R}_{c}$ é a matriz de rotação, dada por

$$ {}^{r}\mathbf{R}_{c} = \begin{bmatrix} cos(\alpha) & 0 & -sen(\alpha) \\ 0 & -1 & 0 \\ -sen(\alpha) & 0 & -\cos(\alpha) \end{bmatrix}, \tag{4.26} $$

e ${}^{r}\mathbf{t}_{c}$ é o vetor de translação, dado por

$$ {}^{r}\mathbf{t}_{c} = \begin{bmatrix} d \\ 0 \\ h-e \end{bmatrix}. \tag{4.27} $$

De acordo com a Figura 4.7, para se passar do referencial da câmera *C* para o referencial do quadro *Q*, as seguintes transformações (representadas por matrizes) devem ser executadas, na ordem apresentada:

*a)* rotação de $-\alpha$ no eixo ${}^{c}Y$, dada por

$$ {}^{c}\mathbf{R}_{y,-\alpha} = \begin{bmatrix} cos(-\alpha) & 0 & -sen(-\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ sen(-\alpha) & 0 & cos(-\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} cos(\alpha) & 0 & sen(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ -sen(\alpha) & 0 & cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.28} $$

*b)* translação *h* no eixo ${}^{c}Z$, dada por

$$ {}^{c}\mathbf{T}_{z,h} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.29} $$

*c)* translação *a* no eixo ${}^{c}X$, dada por

$$ {}^{c}\mathbf{T}_{x,a} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.30} $$

*d)* translação -*b* no eixo ${}^{c}Y$ , dada por

$$ {}^{c}\mathbf{T}_{y,-b} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.31} $$

*e)* rotação de 180º no eixo ${}^{c}Y$ , dada por

$$ {}^{c}\mathbf{R}_{y,180^o} = \begin{bmatrix} cos(180^o) & 0 & -sen(180^o) & 0 \\ 0 & 1 & 0 & 0 \\ sen(180^o) & 0 & cos(180^o) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.32} $$

Assim, a matriz de transformação do referencial da câmera para o referencial do quadro é dada pelo produto das matrizes dos itens *a* até *e*, ou seja

$$ {}^{c}\mathbf{M}_{q} = {}^{c}\mathbf{R}_{y,-\alpha}.{}^{c}\mathbf{T}_{z,h}.{}^{c}\mathbf{T}_{x,a}.{}^{c}\mathbf{T}_{y,-b}.{}^{c}\mathbf{R}_{y,180^o}, $$

$$ {}^{c}\mathbf{M}_{q} = \begin{bmatrix} -\cos(\alpha) & 0 & -sen(\alpha) & a.\cos(\alpha)+h.sen(\alpha) \\ 0 & 1 & 0 & -b \\ sen(\alpha) & 0 & -\cos(\alpha) & -a.sen(\alpha)+h.\cos(\alpha) \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.33} $$

Portanto, dado um vetor no sistema de coordenadas homogêneas da câmera ${}^{c}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{c}x & {}^{c}y & {}^{c}z & 1 \end{bmatrix}^{T}$ e um ponto no sistema de coordenadas homogêneas do quadro ${}^{q}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{q}x & {}^{q}y & {}^{q}z & 1 \end{bmatrix}^{T}$, tem-se que

$$ {}^{c}\tilde{\mathbf{p}} = {}^{c}\mathbf{M}_{q}.{}^{q}\tilde{\mathbf{p}}. \tag{4.34} $$

A matriz de transformação ${}^{c}\mathbf{M}_{q}$ pode ser expressa em termos de matriz de rotação e vetor de translação, como

$$ {}^{c}\mathbf{M}_{q} = \left[\begin{array}{c|c} {}^{c}\mathbf{R}_{q} & {}^{c}\mathbf{t}_{q} \\ \hline 0_{3}^{t} & 1 \end{array}\right], \tag{4.35} $$

onde ${}^{c}\mathbf{R}_{q}$ é a matriz de rotação, dada por

$$ {}^{c}\mathbf{R}_{q} = \begin{bmatrix} -\cos(\alpha) & 0 & -sen(\alpha) \\ 0 & 1 & 0 \\ sen(\alpha) & 0 & -\cos(\alpha) \end{bmatrix}, \tag{4.36} $$

e ${}^{c}\mathbf{t}_{q}$ é o vetor de translação, dado por

$$ {}^{c}\mathbf{t}_{q} = \begin{bmatrix} a.\cos(\alpha) + h.sen(\alpha) \\ -b \\ -a.sen(\alpha) + h.\cos(\alpha) \end{bmatrix}. \tag{4.37} $$

Finalmente, dado um vetor no sistema de coordenadas homogêneas do robô ${}^{r}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{r}x & {}^{r}y & {}^{r}z & 1 \end{bmatrix}^{T}$ e um vetor no sistema de coordenadas homogêneas do quadro ${}^{q}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{q}x & {}^{q}y & {}^{q}z & 1 \end{bmatrix}^{T}$, tem-se que

$$ {}^{r}\tilde{\mathbf{p}} = {}^{r}\mathbf{M}_{c}.{}^{c}\tilde{\mathbf{p}}. \tag{4.38} $$

Mas, como

$$ {}^{c}\tilde{\mathbf{p}} = {}^{c}\mathbf{M}_{q}.{}^{q}\tilde{\mathbf{p}}. \tag{4.39} $$

obtém-se que

$$ {}^{r}\tilde{\mathbf{p}} = {}^{r}\mathbf{M}_{c}.{}^{c}\mathbf{M}_{q}.{}^{q}\tilde{\mathbf{p}}. \tag{4.40} $$

Assim, finalmente,

$$ {}^{r}\mathbf{M}_{q} = {}^{r}\mathbf{M}_{c}.{}^{c}\mathbf{M}_{q}, $$

$$ {}^{r}\mathbf{M}_{q} = \begin{bmatrix} cos(\alpha) & 0 & -sen(\alpha) & d \\ 0 & -1 & 0 & 0 \\ -sen(\alpha) & 0 & -\cos(\alpha) & h-e \\ 0 & 0 & 0 & 1 \end{bmatrix}. \begin{bmatrix} -\cos(\alpha) & 0 & -sen(\alpha) & a.\cos(\alpha) + h.sen(\alpha) \\ 0 & 1 & 0 & -b \\ sen(\alpha) & 0 & -\cos(\alpha) & -a.sen(\alpha) + h.\cos(\alpha) \\ 0 & 0 & 0 & 1 \end{bmatrix}, $$

$$ {}^{r}\mathbf{M}_{q} = \begin{bmatrix} -1 & 0 & 0 & a+d \\ 0 & -1 & 0 & b \\ 0 & 0 & 1 & -e \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.41} $$

A Matriz ${}^{r}\mathbf{M}_{q}$ pode ser expressa por transformações de rotações e translações diretamente do sistema de referenciais do robô, *R*, para o sistema do quadro, *Q*, ou seja,

$$ {}^{r}\mathbf{M}_{q} = {}^{r}\mathbf{R}_{z,180^{o}}.{}^{r}\mathbf{T}_{z,-e}.{}^{r}\mathbf{T}_{x,-(a+d)}.{}^{r}\mathbf{T}_{y,-b}, $$

$$ {}^{r}\mathbf{M}_{q} = \begin{bmatrix} -1 & 0 & 0 & a+d \\ 0 & -1 & 0 & b \\ 0 & 0 & 1 & -e \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.42} $$

Esta matriz de transformação ${}^{r}\mathbf{M}_{q}$ é usada no processo de registro de imagens.

## 4.4 Registro de Imagens de Profundidade

A tarefa de registro envolve encontrar os parâmetros de rotação e translação que alinham corretamente vistas sobrepostas de um objeto, de modo a construir uma representação integrada da sua superfície [45].

Quando o registro entre duas imagens está correto, as porções sobrepostas de duas imagens devem se fundir com pequeno erro, pois elas representam a mesma região da superfície [46].

O registro de imagens pode ocorrer em duas situações: quando se conhece a transformação geométrica entre as duas imagens e quando esta transformação é desconhecida. Nesta segunda situação, haveria a necessidade de usar métodos complexos e heurísticos para encontrar a transformação, cujo custo computacional é alto.

O sistema de controle do robô Guará fornece a matriz de transformação (rotação e translação) entre dois passos sucessivos. Portanto, esta matriz é usada para o registro de duas imagens de profundidade sucessivas.

O registro baseado apenas na odometria do robô pode ter incertezas, devido a erros de medição dos sensores do robô e ao escorregamento das patas em empuxo. Logo, o registro é refinado usando o algoritmo iterativo do ponto mais próximo (*ICP*). A Figura 4.8 mostra os referenciais do quadro, do robô e do mundo, necessários para a transformação de imagens de profundidade consecutivas para o registro.

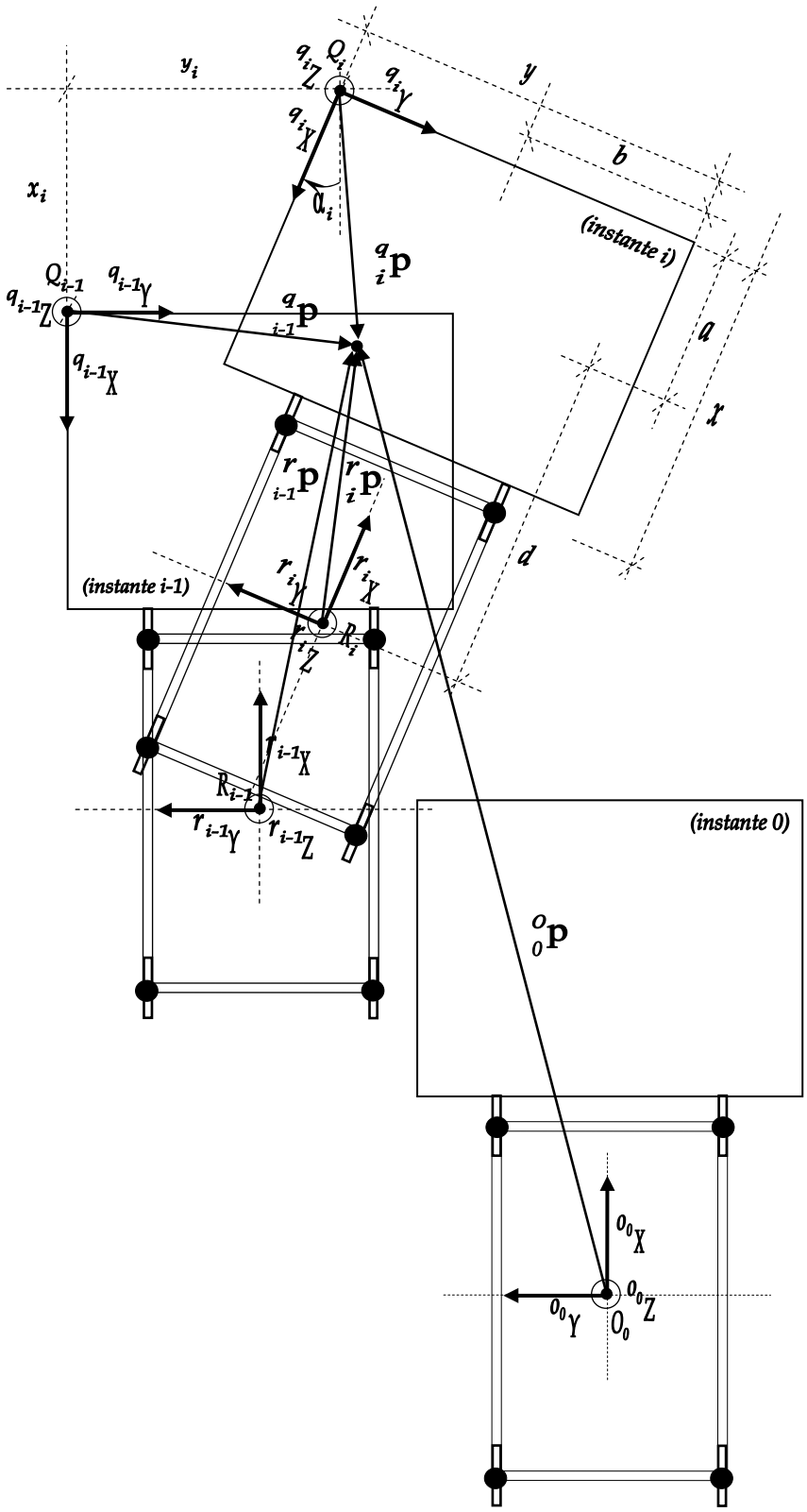


*Figura 4.8: Referenciais para registro de imagens de profundidade consecutivas.*

De acordo com a Figura 4.8, seja $Q_{i-1}$ o referencial da imagem de profundidade no instante $i-1$, e $Q_i$ o referencial da imagem no instante *i*. Para se passar do referencial $Q_{i-1}$ para o referencial $Q_i$, as seguintes transformações devem ser executadas, na ordem apresentada:

*a)* translação $x_i$ no eixo ${}^{q_{i-1}}X$, caracterizada pela matriz

$$ {}^{q}_{i-1}\mathbf{T}_{x,x_i} = \begin{bmatrix} 1 & 0 & 0 & x_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.43} $$

*b)* translação $y_i$ no eixo ${}^{q_{i-1}}Y$, caracterizada pela matriz

$$ {}^{q}_{i-1}\mathbf{T}_{y,y_i} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & y_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{4.44} $$

*c)* rotação de $\alpha_i$ graus no eixo ${}^{q_{i-1}}Z$, caracterizada pela matriz

$$ {}^{q}_{i-1}\mathbf{R}_{z,\alpha_i} = \begin{bmatrix} cos(\alpha_i) & -sen(\alpha_i) & 0 & 0 \\ sen(\alpha_i) & cos(\alpha_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.45} $$

Nota-se que, em relação aos referenciais adotados, $x_i < 0$, $y_i > 0$ e $\alpha_i < 0$. A matriz de transformação do referencial $Q_{i-1}$ para o referencial $Q_i$ é dada pelo produto das matrizes dos itens *a* até *c*, ou seja, por

$$ {}^{q}_{i-1}\mathbf{M}^{i}_{q} = {}^{q}_{i-1}\mathbf{T}_{x,x_i} \cdot {}^{q}_{i-1}\mathbf{T}_{y,y_i} \cdot {}^{q}_{i-1}\mathbf{R}_{z,\alpha_i}, $$

$$ {}^{q}_{i-1}\mathbf{M}^{i}_{q} = \begin{bmatrix} \cos(\alpha_i) & -sen(\alpha_i) & 0 & x_i \\ sen(\alpha_i) & \cos(\alpha_i) & 0 & y_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.46} $$

Portanto, dado um ponto ${}^{q}_{i-1}\mathbf{p}$ no referencial $Q_{i-1}$, cujo vetor homogêneo é ${}^{q}_{i-1}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{q}_{i-1}x & {}^{q}_{i-1}y & {}^{q}_{i-1}z & 1 \end{bmatrix}^T$, e o mesmo ponto ${}^{q}_{i}\mathbf{p}$ no referencial $Q_i$, cujo vetor homogêneo é ${}^{q}_{i}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{q}_{i}x & {}^{q}_{i}y & {}^{q}_{i}z & 1 \end{bmatrix}^T$, tem-se que

$$ {}^{q}_{i-1}\tilde{\mathbf{p}} = {}^{q}_{i-1}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}. \tag{4.47} $$

Desta maneira, os *pixels* da imagem de profundidade $i$ devem sofrer as transformações geométricas mostradas em (4.46), e serem integrados à imagem de profundidade $i-1$. O resultado é uma imagem única, integrando os dois quadros. O processo se repete para os quadros sucessivos, de modo que, para $n$ quadros, tem-se que

$$ \begin{aligned} {}^{q}_{0}\tilde{\mathbf{p}} &= {}^{q}_{0}\mathbf{M}^{1}_{q} \cdot {}^{q}_{1}\mathbf{M}^{2}_{q} \cdots {}^{q}_{i-1}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}, \\ {}^{q}_{0}\tilde{\mathbf{p}} &= {}^{q}_{0}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}. \end{aligned} \tag{4.48} $$

A Figura 4.9 mostra a imagem de profundidade em relação ao referencial $Q_0$, a imagem de profundidade em relação ao referencial $Q_1$ e a imagem de profundidade integrada em relação ao referencial $Q_0$. Observe-se que há áreas da superfície comuns aos dois quadros.

A imagem de profundidade integrada em $Q_0$ contém pontos não mapeados pela transformação. Nesta imagem não foi aplicada a transformação bi-linear sobre os pontos não mapeados porque ela ainda sofrerá outras transformações posteriormente.

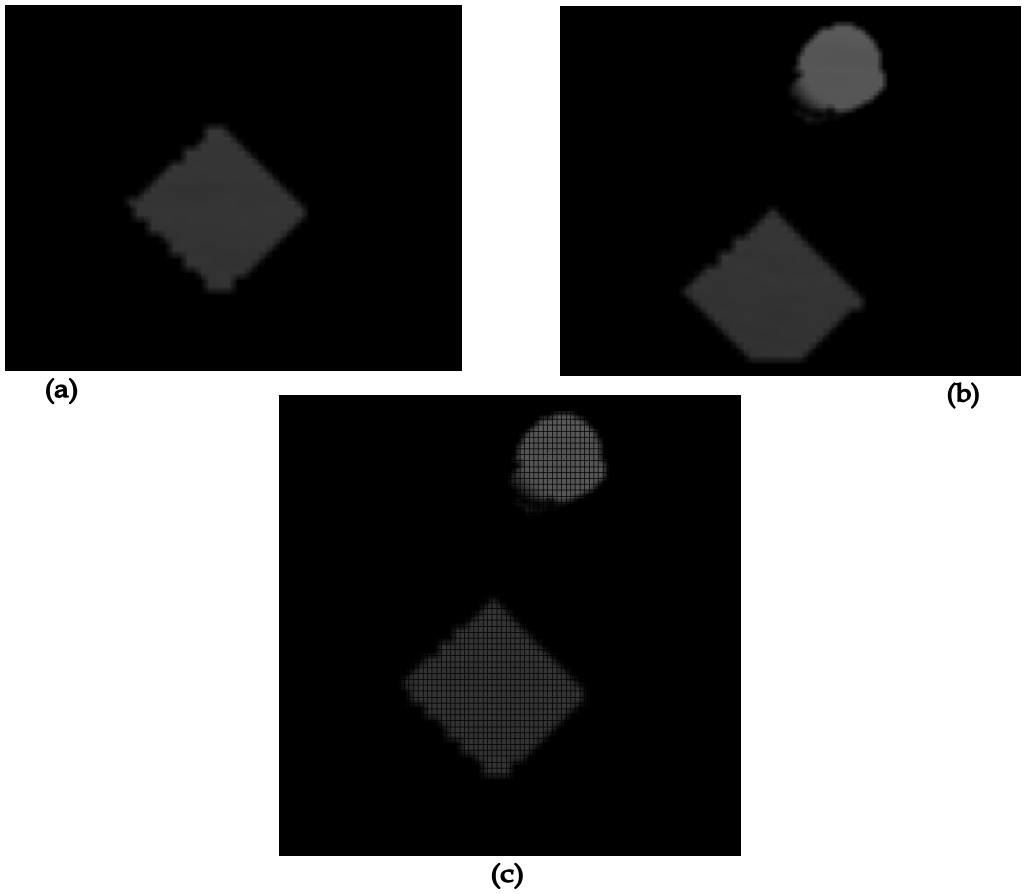


*Figura 4.9: (a) Imagem de profundidade em $Q_0$, (b) Imagem de profundidade em $Q_1$ e (c) Imagem de profundidade integrada em $Q_0$.*

Para o controle do sistema de navegação do robô Guará é necessário que os pontos sejam mapeados em coordenadas no referencial do robô e também no referencial do mundo.

A matriz de transformação do robô em relação ao referencial do mundo, descrita em (4.18), é fornecida pelo sistema de controle do robô, a cada novo passo.

Assim, dado um ponto no referencial do quadro no instante *i*, ${}^{q}_{i}\tilde{\mathbf{p}}$, ele é convertido para o referencial do robô pela transformação

$$ {}^{r}_{i}\tilde{\mathbf{p}} = {}^{r}_{i}\mathbf{M}^{i}_{q}.{}^{q}_{i}\tilde{\mathbf{p}}. \tag{4.49}$$

Nos instantes de análise *i-1* e *i*, os vetores homogêneos ${}^{r}_{i-1}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{r}_{i-1}x & {}^{r}_{i-1}y & {}^{r}_{i-1}z & 1 \end{bmatrix}^T$ e ${}^{r}_{i}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{r}_{i}x & {}^{r}_{i}y & {}^{r}_{i}z & 1 \end{bmatrix}^T$, cujos respectivos pontos são ${}^{r}_{i-1}\mathbf{p}$ e ${}^{r}_{i}\mathbf{p}$, são representados pela transformação

$$ {}^{r}_{i-1}\tilde{\mathbf{p}} = {}^{r}_{i-1}\mathbf{M}^{i}_{r} \cdot {}^{r}_{i}\tilde{\mathbf{p}}. \tag{4.50} $$

Generalizando para *n* passos completados pelo robô e iniciando no instante 0, tem-se que

$$ \begin{aligned} {}^{r}_{0}\tilde{\mathbf{p}} &= {}^{r}_{0}\mathbf{M}^{1}_{r} \cdot {}^{r}_{1}\mathbf{M}^{2}_{r} \cdots {}^{r}_{i-1}\mathbf{M}^{i}_{r} \cdot {}^{r}_{i}\tilde{\mathbf{p}}, \\ {}^{r}_{0}\tilde{\mathbf{p}} &= {}^{r}_{0}\mathbf{M}^{i}_{r} \cdot {}^{r}_{i}\tilde{\mathbf{p}}. \end{aligned} \tag{4.51} $$

A conversão do vetor ${}^{r}_{0}\tilde{\mathbf{p}}$ para o referencial do mundo, no instante 0, é dada por

$$ {}^{o}_{0}\tilde{\mathbf{p}} = {}^{o}_{0}\mathbf{M}^{0}_{r} \cdot {}^{r}_{0}\tilde{\mathbf{p}}, \tag{4.52} $$

onde ${}^{o}_{0}\tilde{\mathbf{p}} = \begin{bmatrix} {}^{o}_{0}x & {}^{o}_{0}y & {}^{o}_{0}z & 1 \end{bmatrix}^T$ é o vetor homogêneo de ${}^{o}_{0}\mathbf{p}$, que representa o ponto de análise em coordenadas do mundo em relação ao instante 0. Neste instante 0, o sistema de coordenadas do mundo e do robô estão alinhados, separados apenas pela elevação *e*.

Portanto, a relação entre ${}^{o}_{0}\tilde{\mathbf{p}}$ e ${}^{q}_{i}\tilde{\mathbf{p}}$ é dada por

$$ {}^{o}_{0}\tilde{\mathbf{p}} = {}^{o}_{0}\mathbf{M}^{0}_{r} \cdot {}^{r}_{0}\mathbf{M}^{i}_{r} \cdot {}^{r}_{i}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}. \tag{4.53} $$

Para cada quadro gerado, a matriz de transformação ${}^{r}_{i}\mathbf{M}^{i}_{q}$ é constante, independente do instante da análise, pois os referenciais do robô e do quadro caminham juntos, sem movimento relativo entre eles. A matriz de transformação dada pela equação (4.18) é ${}^{o}_{0}\mathbf{M}^{i}_{r} = {}^{o}_{0}\mathbf{M}^{0}_{r} \cdot {}^{r}_{0}\mathbf{M}^{i}_{r}$, portanto tem-se, a partir de (4.53) que

$$ \begin{aligned} {}^{o}_{0}\tilde{\mathbf{p}} &= {}^{o}_{0}\mathbf{M}^{i}_{r} \cdot {}^{r}_{i}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}, \\ {}^{o}_{0}\tilde{\mathbf{p}} &= {}^{o}_{0}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}. \end{aligned} \tag{4.54} $$

Esta matriz de transformação, ${}^{o}_{0}\mathbf{M}^{i}_{q}$, é aplicada para todos os pontos da imagem de profundidade do instante *i*, e os pontos transformados são integrados aos pontos da imagem de profundidade do instante *i-1*. Posteriormente, os pontos redundantes das áreas comuns às duas imagens são eliminados no processo de cubificação.

## 4.5 Refinamento do Registro de Imagens pelo Algoritmo *ICP*

Foi necessário elaborar um algoritmo que compensasse o erro de odometria do robô, aumentando assim a precisão do registro de imagens, e, conseqüentemente, garantindo uma navegação mais segura em ambientes com obstáculos.

A solução proposta foi o uso do Algoritmo Iterativo do Ponto Mais Próximo, do inglês *Iterative Closest Point* (*ICP*), que é usado para corrigir o erro de alinhamento de imagens quando há pontos correspondentes entre duas imagens que contenham áreas comuns.

O objetivo do algoritmo é minimizar a soma dos erros quadráticos em relação aos pontos mais próximos da imagem 1 e seus pontos correspondentes da imagem 2.

É importante salientar que uma estimativa inicial da posição para alinhamento das duas imagens deve ser feita. Desta maneira, uma transformação (rotação e translação) apropriada na imagem 1 deve ser aplicada, baseada nesta estimativa inicial, para alinhar as imagens de forma aproximada. A matriz de transformação do robô Guará, usada para o registro de imagens, é usada para este fim. Esta matriz indica a rotação e translação do referencial *R* do robô em relação ao referencial *O* do mundo, acumulada desde o instante 0 de análise.

O algoritmo calcula a menor distância entre cada ponto da imagem 1 e o ponto correspondente na imagem 2. Estes pontos calculados são, então, usados para criar uma matriz de transformação que será aplicada a todos os pontos da imagem 1 para ajustá-los na direção da imagem 2. Este processo é repetido várias vezes, de forma iterativa.

Como resultado, os pontos da imagem 1 são deslocados para a região onde há menor distância com os pontos correspondentes da imagem 2. O algoritmo tem convergência quando o erro médio das menores distâncias entre os pontos correspondentes de duas iterações sucessivas fica abaixo de um valor de tolerância específico. Neste momento, o algoritmo pára o processo de iteração.

Com o *ICP*, o alinhamento das imagens melhora significativamente, reduzindo o erro. A matriz de rotação e translação gerada pelo *ICP* é usada para realimentar a odometria do robô, compensando eventuais erros da trajetória.

O *ICP* usa imagens de profundidade de quadros sucessivos. Este algoritmo tem boa convergência caso as áreas de mapeamento comuns das duas imagens sejam bem definidas, com informação de profundidade variada. Imagens com superfícies planas não têm informação de profundidade variada, e tendem a fazer com que o algoritmo não tenha boa convergência.

Rusinkiewicz e Levoy [47] fizeram um estudo minucioso de comparação de várias variantes do algoritmo *ICP*, onde os critérios de desempenho, seleção de pontos próximos correspondentes e a convergência dos algoritmos são comparados.

### 4.5.1 Técnica Criada para Melhor Convergência do *ICP*

Para aumentar o grau de convergência do algoritmo *ICP*, e conseqüentemente aumentar a precisão do registro de imagens, vários testes foram feitos com obstáculos de diferentes formas e tamanhos. Notou-se experimentalmente que para objetos esguios, com uma definição preferencial de sua direção, o *ICP* apresenta melhor convergência do que para objetos redondos.

Como forma de reduzir o tempo de processamento do algoritmo *ICP*, a técnica criada reduz o universo de pontos válidos correspondentes, escolhendo-os de forma eficiente e criteriosa. A seguinte solução foi implementada:

- ✓ binarização da imagem 1;
- ✓ esqueletização da imagem binarizada;
- ✓ interseção da imagem esqueletizada com a imagem 1 original.

A binarização segmenta a imagem, separando os objetos. O algoritmo de esqueletização mostrado no capítulo 4 foi modificado para que, além da varredura vertical dos *pixels*,

também fizesse a varredura horizontal, para que os esqueletos dos objetos fossem obtidos em duas direções preferenciais.

Este algoritmo de esqueletização, por ser uma simplificação do algoritmo de esqueletização iterativa que usa operações de erosão (*erosion*) e abertura (*opening*) aplicadas à imagem, tem baixo custo computacional.

Finalmente, a informação de profundidade é obtida com a interseção da imagem esqueletizada monocromática com a imagem original. Para cada *pixel* branco da imagem esqueletizada, o *pixel* correspondente da imagem original é armazenado, guardando, assim, a informação de profundidade.

A Figura 4.10 mostra um exemplo de imagem 1 e o seu tratamento para uso no ICP.

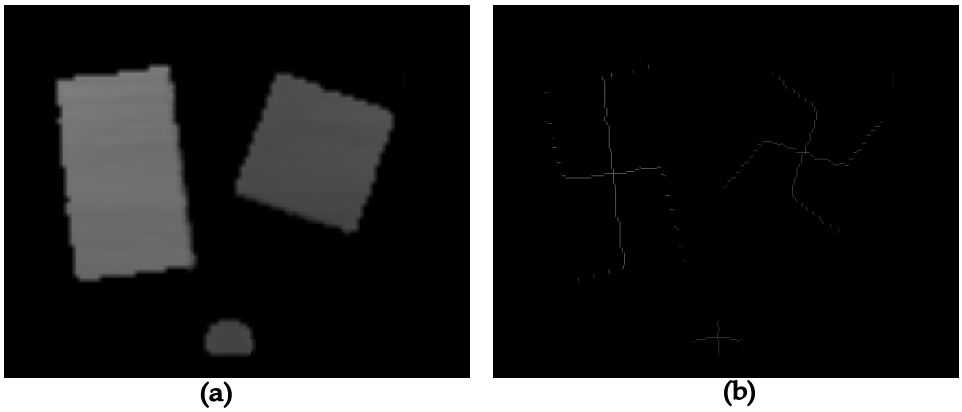


*Figura 4.10: (a) Imagem original e (b) Imagem tratada para o ICP.*

### 4.5.2 Estimativa Inicial

Uma boa estimativa inicial garante uma rápida convergência do algoritmo, reduzindo a quantidade de iterações e tornando-o mais rápido. Neste caso, a informação da odometria do robô, através da matriz de rotação e translação entre dois quadros sucessivos, é fornecida ao algoritmo como estimativa inicial para alinhamento das imagens.

### 4.5.3 Implementação Proposta

Um dos passos mais importantes do *ICP* é calcular a menor distância entre cada ponto válido da imagem 1 e o ponto correspondente na imagem 2. Os pontos são representados em coordenadas *3D*. A distância euclidiana entre os pontos é representada por

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}, \tag{4.55}$$

onde $(x_1, y_1, z_1)$ é um ponto da imagem 1 e $(x_2, y_2, z_2)$ é um ponto da imagem 2. Este cálculo é executado para cada ponto válido da imagem 1 contra todos os pontos válidos da imagem 2. A coordenada $(x_2, y_2, z_2)$ que produzir o menor valor de distância é armazenada como ponto mais próximo de $(x_1, y_1, z_1)$.

O método implementado é um algoritmo baseado em quarteniões descrito por Besh e McKay [48] para registro de formas *3D*. Este método foi adaptado para registro de imagens de profundidade.

A matriz de transformação **Tr** (rotação e translação) que mapeia as áreas comuns entre a imagem 1 e a imagem 2 é obtida com a ajuda de quarteniões. Os quarteniões são generalizações de números complexos.

Rotações espaciais podem ser parametrizadas pelos ângulos de Euler $(\phi, \theta, \psi)$ e por quarteniões unitários $\mathbf{q} = [q_0 \quad q_1 \quad q_2 \quad q_3]^T$. Um quartenião unitário é descrito por um vetor unitário com a forma $\mathbf{q}^t.\mathbf{q} = q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. A vantagem de quarteniões é que, computacionalmente, eles são mais eficientes do que usar os ângulos de Euler, que exigem funções trigonométricas que apresentam pontos de singularidade. Os 3 ângulos de Euler podem ser substituídos por 4 quarteniões unitários, que apresentam apenas operações de adições e multiplicações.

Seja $\mathbf{p_{1k}} = \begin{bmatrix} x_{1k} & y_{1k} & z_{1k} \end{bmatrix}^T$ um ponto (vetor) em análise na imagem 1, para $k = 1, 2, 3, ..., n$, onde $n$ indica a quantidade de pontos válidos. De forma análoga, seja $\mathbf{p_{2k}} = \begin{bmatrix} x_{2k} & y_{2k} & z_{2k} \end{bmatrix}^T$ o ponto correspondente a $\mathbf{p_{1k}}$ na imagem 2. A função objetivo dos mínimos quadrados a ser minimizada é

$$f(\mathbf{Q}) = \frac{1}{n}\sum_{k=1}^{n} \left\| \mathbf{p_{2k}} - \mathbf{Tr(Q)}.\mathbf{p_{1k}} \right\|, \tag{4.56}$$

onde $\mathbf{Q}$ é a matriz quartenião e $\mathbf{Tr}$ é a matriz de transformação.

É necessário calcular o valor médio (centro de massa) $\mathbf{m}_1$ dos pontos válidos da imagem 1 e o valor médio (centro de massa) $\mathbf{m}_2$ dos pontos válidos da imagem 2, conforme

$$\begin{aligned} \mathbf{m}_1 &= \left[ \frac{1}{n}\sum_{k=1}^{n} x_{1k} \quad \frac{1}{n}\sum_{k=1}^{n} y_{1k} \quad \frac{1}{n}\sum_{k=1}^{n} z_{1k} \right]^T, \\ \mathbf{m}_2 &= \left[ \frac{1}{n}\sum_{k=1}^{n} x_{2k} \quad \frac{1}{n}\sum_{k=1}^{n} y_{2k} \quad \frac{1}{n}\sum_{k=1}^{n} z_{2k} \right]^T. \end{aligned} \tag{4.57}$$

A *variância* de um vetor $\mathbf{x} = [x_1 \quad x_2 \quad ... \quad x_n]^T$ escreve-se por $\sigma_x^2$, e mede o afastamento dos seus elementos em torno da sua média $\mu_x$. A raiz quadrada da variância designa-se pelo desvio padrão $\sigma_x$. A *variância* é dada por

$$\sigma_x^2 = \frac{(\mathbf{x} - \mu_x)^T (\mathbf{x} - \mu_x)}{(n-1)}. \tag{4.58}$$

A *covariância* calcula-se de forma análoga à variância, mas envolve dois vetores de igual dimensão. A covariância entre o vetor $\mathbf{x}$ e o vetor $\mathbf{y}$ de dimensão *n* calcula-se de acordo com a equação

$$\sigma_{xy}^2 = \frac{(\mathbf{x} - \mu_x)^T (\mathbf{y} - \mu_y)}{(n-1)}. \tag{4.59}$$

Em termos gerais, define-se a *matriz de covariância* como a matriz que contém na sua diagonal principal as variâncias das colunas e nos demais elementos as covariâncias.

Utilizando todos os pontos válidos de ambas as imagens, a matriz de covariância é dada por

$$\mathbf{C} = \frac{1}{n}\sum_{k=1}^{n}\left(\mathbf{p}_{1k}.\mathbf{p}_{2k}{}^{T} - \mathbf{m}_1.\mathbf{m}_2{}^{T}\right). \tag{4.60}$$

Uma vez que a matriz de covariância é calculada, ela é usada no cálculo da matriz de quartenião. Seja a matriz de covariância $\mathbf{C}$ e seus respectivos coeficientes, ou seja,

$$\mathbf{C} = \begin{bmatrix} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{bmatrix}, \tag{4.61}$$

e seja $\mathbf{I}$ a matriz identidade 3x3:

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.62}$$

define-se $\mathbf{A}$ como segue:

$$\mathbf{A} = \mathbf{C} - \mathbf{C}^{T} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{bmatrix}, \tag{4.63}$$

e define-se $\mathbf{d}$, um vetor linha, como

$$\mathbf{d} = [a_6 \quad a_7 \quad a_2]. \tag{4.64}$$

Seja $s = c_1 + c_5 + c_9$ um escalar, e a matriz $\mathbf{T}$ é

$$\mathbf{T} = \mathbf{C} + \mathbf{C}^{T} - s.\mathbf{I} = \begin{bmatrix} t_1 & t_2 & t_3 \\ t_4 & t_5 & t_6 \\ t_7 & t_8 & t_9 \end{bmatrix}. \tag{4.65}$$

Conseqüentemente, a matriz de quartenião $\mathbf{Q}$ torna-se

$$\mathbf{Q} = \begin{bmatrix} s & a_6 & a_7 & a_2 \\ a_6 & t_1 & t_2 & t_3 \\ a_7 & t_4 & t_5 & t_6 \\ a_2 & t_7 & t_8 & t_9 \end{bmatrix}. \tag{4.66}$$

O passo final do algoritmo *ICP* é o cálculo da matriz de transformação (rotação e translação em notação homogênea). O primeiro passo é encontrar o máximo auto-valor e seu correspondente auto-vetor para a matriz de quartenião $\mathbf{Q}$. Defina-se o vetor com os auto-valores na forma de vetor linha $\mathbf{q} = [q_1 \quad q_2 \quad q_3 \quad q_4]$. Seja $\tilde{\mathbf{m}}_1 = \left[\overline{x_1} \quad \overline{y_1} \quad \overline{z_1} \quad 1\right]^t$ o vetor

média dos pontos da imagem 1 em coordenadas homogêneas e $\tilde{\mathbf{m}}_2 = \begin{bmatrix} \overline{x_2} & \overline{y_2} & \overline{z_2} & 1 \end{bmatrix}^T$ o vetor média dos pontos equivalentes da imagem 2 em coordenadas homogêneas. Assim, é possível definir a matriz de transformação como

$$\mathbf{Tr} = \begin{bmatrix} tr_1 & tr_2 & tr_3 & tr_4 \\ tr_5 & tr_6 & tr_7 & tr_8 \\ tr_9 & tr_{10} & tr_{11} & tr_{12} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.67}$$

onde

$$\begin{aligned} tr_1 &= q_1^2 + q_2^2 - q_3^2 - q_4^2, \\ tr_2 &= 2.(q_2.q_3 - q_1.q_4), \\ tr_3 &= 2.(q_2.q_4 + q_1.q_3), \\ tr_4 &= 0, \\ tr_5 &= 2.(q_2.q_3 + q_1.q_4), \\ tr_6 &= q_1^2 + q_3^2 - q_2^2 - q_4^2, \\ tr_7 &= 2.(q_3.q_4 + q_1.q_2), \\ tr_8 &= 0, \\ tr_9 &= 2.(q_2.q_4 - q_1.q_3), \\ tr_{10} &= 2.(q_3.q_4 + q_1.q_2), \\ tr_{11} &= q_1^2 + q_4^2 - q_2^2 - q_3^2, \\ tr_{12} &= 0. \end{aligned} \tag{4.68}$$

A matriz **Tr** está apenas calculando a rotação que a imagem 1 deve fazer. Para calcular a translação, o seguinte passo deve ser feito:

defina o vetor **l** como

$$\mathbf{l} = \tilde{\mathbf{m}}_2 - \mathbf{Tr}.\tilde{\mathbf{m}}_1 = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \end{bmatrix}, \tag{4.69}$$

e atualize

$$\begin{aligned} tr_4 &= l_1, \\ tr_8 &= l_2, \\ tr_{12} &= l_3. \end{aligned} \tag{4.70}$$

De posse da matriz de transformação, todos os pontos da imagem 1 são atualizados, multiplicando-os por esta matriz. Assim, todo o processo é repetido (nova iteração) até que

os pontos válidos da imagem 1 estejam suficientemente próximos aos seus correspondentes da imagem 2.

O processo pára quando a distância média euclidiana dos pontos válidos entre as duas imagens estiver dentro de uma faixa estreita de valor (*threshold*), ou seja, minimizando a função objetivo dos mínimos quadrados $f(\mathbf{Q})$.

### 4.5.5 Resultados

Os resultados se mostraram bastante satisfatórios graças à escolha criteriosa dos pontos válidos da imagem 1. O algoritmo apresenta convergência quando há áreas comuns nas duas imagens, o que ocorre durante a aquisição das imagens.

Sejam duas imagens adquiridas em passos sucessivos do robô. A Figura 4.11(a) mostra a imagem 1 adquirida no instante $i=3$ e a Figura 4.11(b) mostra a imagem 2 adquirida no instante $i= 2$.

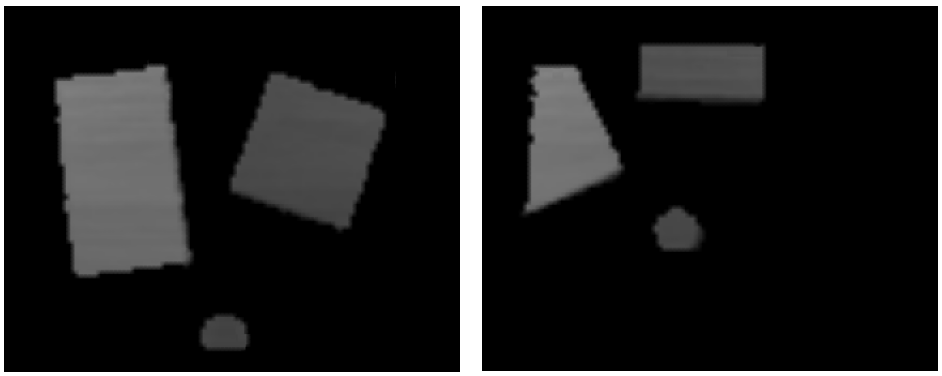


*Figura 4.11: (a) Imagem 1 (b) Imagem 2.*

A Figura 4.12 mostra o registro das duas imagens da Figura 4.11. Na Figura 4.12(a), o registro foi feito levando-se em conta apenas a matriz de rotação e translação fornecida pela odometria do robô. Na Figura 4.12(b), a informação da odometria é usada como estimativa inicial e o algoritmo *ICP* corrige o erro, melhorando substancialmente o registro das

imagens e gerando uma matriz de transformação que é enviada ao robô para correção da sua trajetória.

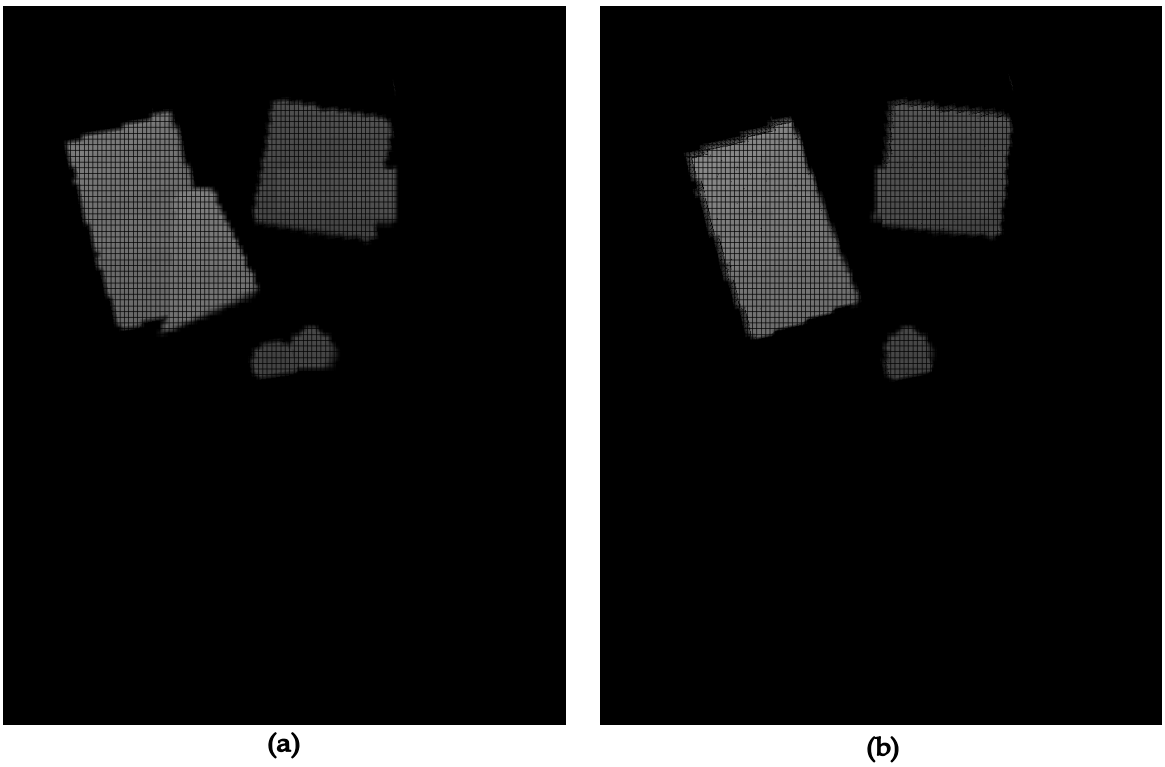


*Figura 4.12: (a) Registro de imagens com a odometria (b) Registro de imagens com a odometria e o ICP.*

Para a Figura 4.12(a), a matriz de transformação da odometria ${}^{o}_{0}\mathbf{M}^{3}_{q}$ para registro da imagem 1 sobre a imagem 2 em coordenadas do mundo, para o instante 3 é

$${}^{o}_{0}\mathbf{M}^{3}_{q} = \begin{bmatrix} -0{,}985 & 0{,}174 & 0{,}000 & 1136{,}341 \\ -0{,}174 & -0{,}985 & 0{,}000 & 410{,}475 \\ 0{,}000 & 0{,}000 & 1{,}000 & 9{,}000 \\ 0{,}000 & 0{,}000 & 0{,}000 & 1{,}000 \end{bmatrix}. \tag{4.71}$$

Para a Figura 4.12(b), a matriz de transformação do *ICP* ${}^{o}_{3}\mathbf{Tr}^{3}_{o}$ para refinamento do registro da imagem 1 sobre a imagem 2, em coordenadas do mundo, para o instante 3 é

$${}^{o}_{3}\mathbf{Tr}^{3}_{o} = \begin{bmatrix} 0{,}998 & -0{,}054 & -0{,}025 & 6{,}772 \\ 0{,}053 & 0{,}998 & -0{,}034 & -83{,}500 \\ 0{,}027 & 0{,}033 & 0{,}999 & -24{,}079 \\ 0{,}000 & 0{,}000 & 0{,}000 & 1{,}000 \end{bmatrix}. \tag{4.72}$$

Logo, a matriz de transformação final ${}^{o}_{0}\mathbf{M}^{3'}_{q}$, com o registro e o refinamento pelo *ICP* da Figura 4.12(b), é dada por

$$ {}^{o}_{0}\mathbf{M}^{3'}_{q} = {}^{o}_{3}\mathbf{Tr}^{3}_{o} \cdot {}^{o}_{0}\mathbf{M}^{3}_{q} = \begin{bmatrix} -0,976 & 0,227 & -0,025 & 1118,450 \\ -0,226 & -0,974 & -0,034 & 386,074 \\ -0,032 & -0,028 & 0,999 & 29,139 \\ 0,000 & 0,000 & 0,000 & 1,000 \end{bmatrix}. \tag{4.73} $$

## 4.6 Correção da Trajetória do Robô pelo Algoritmo *ICP*

Graças ao *ICP*, uma matriz de transformação é gerada entre dois quadros sucessivos. Esta informação relativa é usada para corrigir a trajetória do robô. Contudo, só é possível usar esta matriz quando houver duas imagens sucessivas que tenham regiões comuns de obstáculos mapeados, pois neste caso há a convergência do algoritmo ICP.

A Figura 4.13 mostra os referenciais do sistema, onde $Q'_i$ representa o referencial $Q_i$ corrigido pelo algoritmo *ICP*.

Pela equação (4.53), cada ponto ${}^{q}_{i}\mathbf{p}$ do quadro $Q_i$, com seu vetor homogêneo ${}^{q}_{i}\tilde{\mathbf{p}}$, é convertido no correspondente ponto ${}^{o}_{0}\mathbf{p}$, cujo vetor homogêneo é ${}^{o}_{0}\tilde{\mathbf{p}}$. O algoritmo *ICP* foi ajustado para processar em função das coordenadas do mundo $O_0$. O vetor ${}^{o}_{0}\tilde{\mathbf{p}}$ representa a estimativa inicial do *ICP* e o ponto ${}^{o}_{0}\tilde{\mathbf{p}}'$ é o ponto corrigido pela transformação.

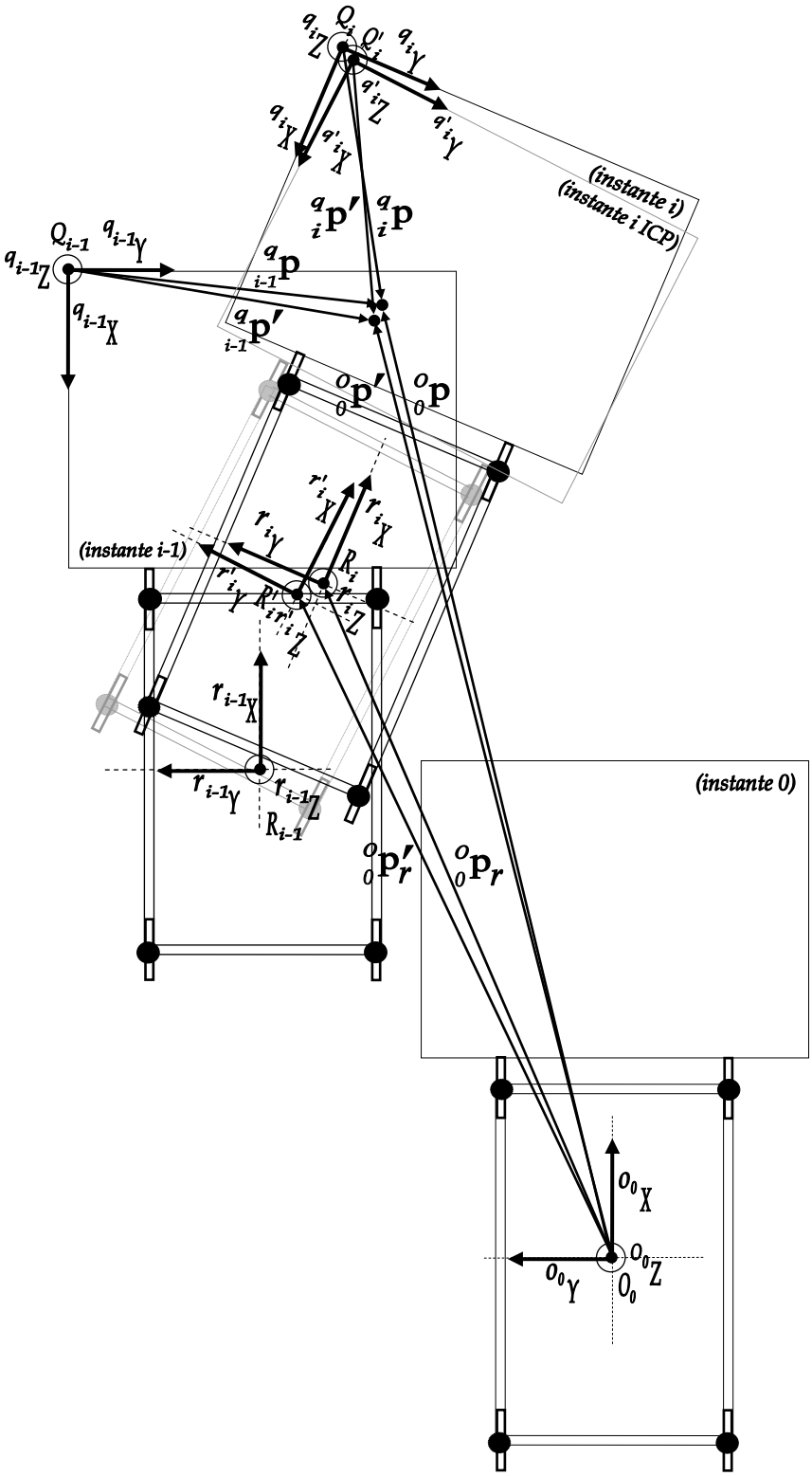


*Figura 4.13: Correção de posição dos pontos e referenciais pelo algoritmo ICP.*

Seja

$$ {}^{o}_{i}\mathbf{Tr}^{i}_{o} = \begin{bmatrix} tr_1 & tr_2 & tr_3 & tr_4 \\ tr_5 & tr_6 & tr_7 & tr_8 \\ tr_9 & tr_{10} & tr_{11} & tr_{12} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.74} $$

a matriz de transformação obtida pela correção de alinhamento de duas imagens sucessivas, no instante *i*, e em relação às coordenadas do mundo, usando o algoritmo *ICP*. Assim,

$$ {}^{o}_{0}\tilde{\mathbf{p}}' = {}^{o}_{i}\mathbf{Tr}^{i}_{o} \cdot {}^{o}_{0}\tilde{\mathbf{p}}. \tag{4.75} $$

Note que quanto menor for o erro de odometria, mais próximo da matriz identidade a matriz ${}^{o}_{i}\mathbf{Tr}^{i}_{o}$ será. Como o ponto ${}^{o}_{0}\tilde{\mathbf{p}} = {}^{o}_{0}\mathbf{M}^{0}_{r} \cdot {}^{r}_{0}\mathbf{M}^{i}_{r} \cdot {}^{r}_{i}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}$, tem-se que

$$ {}^{o}_{0}\tilde{\mathbf{p}}' = {}^{o}_{i}\mathbf{Tr}^{i}_{o} \cdot {}^{o}_{0}\mathbf{M}^{0}_{r} \cdot {}^{r}_{0}\mathbf{M}^{i}_{r} \cdot {}^{r}_{i}\mathbf{M}^{i}_{q} \cdot {}^{q}_{i}\tilde{\mathbf{p}}, \tag{4.76} $$

onde vetor homogêneo ${}^{o}_{0}\tilde{\mathbf{p}}' = \begin{bmatrix} {}^{o}x' & {}^{o}y' & {}^{o}z' & 1 \end{bmatrix}^T$ corresponde ao ponto ${}^{o}_{0}\mathbf{p}'$.O sistema de controle do robô recebe a matriz ${}^{o}_{i}\mathbf{Tr}^{i}_{o}$ e corrige a trajetória, de forma que

$$ {}^{o}_{0}\mathbf{M}^{i\prime}_{r} = {}^{o}_{i}\mathbf{Tr}^{i}_{o} \cdot {}^{o}_{0}\mathbf{M}^{i}_{r}. \tag{4.77} $$

A matriz de transformação acumulada ${}^{o}_{0}\mathbf{M}^{i\prime}_{r}$ passa a ser usada para geração de dados de odometria do próximo passo do robô, a qual é dada por

$$ {}^{o}_{0}\mathbf{M}^{i\prime}_{r} = \left[\begin{array}{ccc:c} r_{11}' & r_{12}' & r_{13}' & x_r' \\ r_{21}' & r_{22}' & r_{23}' & y_r' \\ r_{31}' & r_{32}' & r_{33}' & z_r' \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right]. \tag{4.78} $$

Seja o vetor homogêneo ${}^{o}_{0}\tilde{\mathbf{p}}_r = \begin{bmatrix} x_r & y_r & z_r & 1 \end{bmatrix}^T$ correspondente ao centro geométrico do robô ${}^{o}_{0}\mathbf{p}_r$, em relação à origem. Após a correção, este vetor homogêneo passa a ser ${}^{o}_{0}\tilde{\mathbf{p}}_r' = \begin{bmatrix} x_r' & y_r' & z_r' & 1 \end{bmatrix}^T$, correspondente ao ponto ${}^{o}_{0}\mathbf{p}_r'$.

Seja a matriz acumulada de rotação da plataforma do robô, no instante *i* em relação à origem ${}^{o}_{0}\mathbf{R}^{i}_{mnp} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$. Após a correção, esta matriz passa a ser ${}^{o}_{0}\mathbf{R}^{i}_{mnp}{}' = \begin{bmatrix} r_{11}' & r_{12}' & r_{13}' \\ r_{21}' & r_{22}' & r_{23}' \\ r_{31}' & r_{32}' & r_{33}' \end{bmatrix}$.

Caso o algoritmo *ICP* não tenha convergência, a matriz ${}^{o}_{i}\mathbf{Tr}^{i}_{o} = \mathrm{I}_{4x4}$ e a matriz ${}^{o}_{0}\mathbf{M}^{i\prime}_{r} = {}^{o}_{0}\mathbf{M}^{i}_{r}$, ou seja, ${}^{o}_{0}\tilde{\mathbf{p}}_{r}{}' = {}^{o}_{0}\tilde{\mathbf{p}}_{r}$ e ${}^{o}_{0}\mathbf{R}^{i}_{mnp}{}' = {}^{o}_{0}\mathbf{R}^{i}_{mnp}$. Somente os dados da odometria são usados na integração de imagens e não há refinamento do alinhamento pelo *ICP*.

## 4.7 Modelagem da Andadura do Robô Guará

Uma *andadura* do robô é caracterizada pela seqüência periódica de movimentos que representam a evolução do estado das patas, passando de um estado em apoio a um estado em vôo, quando a pata está sendo transferida para um novo ponto de apoio, e novamente em apoio.

Uma passada ou ciclo de locomoção do robô é um ciclo completo dos movimentos das pernas do robô, tomando-se como referência o momento em que a pata dianteira esquerda é apoiada.

A *fase de transferência* da perna é o período no qual a pata está em vôo para um novo empuxo; o estado de uma perna em transferência é *0*. A *fase de apoio* da perna é o período no qual a pata correspondente está em empuxo; o estado de uma perna em apoio é *1*. O *tempo total de ciclo T* é o tempo para um ciclo completo de locomoção de uma pata em um passo periódico [2].

Seja $T_{ai}$ o tempo da fase de apoio da pata *i* e $T_i$ o tempo total do ciclo da pata *i*. Logo, $\beta_i$ é definido como o fator de carga da pata *i*, pela razão entre o tempo da fase de apoio e o tempo total de ciclo *T*, ou seja,

$$\beta = \frac{T_{ai}}{T_i} \tag{4.79}$$

A andadura escolhida para o robô Guará é a *regular simétrica em onda periódica de ciclo completo não singular* [2]. Nesta andadura, os eventos de vôo para novo posicionamento e empuxo são simétricos e estão distribuídos em um ciclo, sendo que os eventos de vôo estão na seqüência das patas *3*, 2, *1* e *0*, iniciando com a pata traseira direita, *3*, e terminando com a pata dianteira esquerda *0*, conforme mostrado no diagrama da Figura 4.14. A pata no solo é mostrada pela linha em negrito e a pata em vôo onde há descontinuidade da linha.

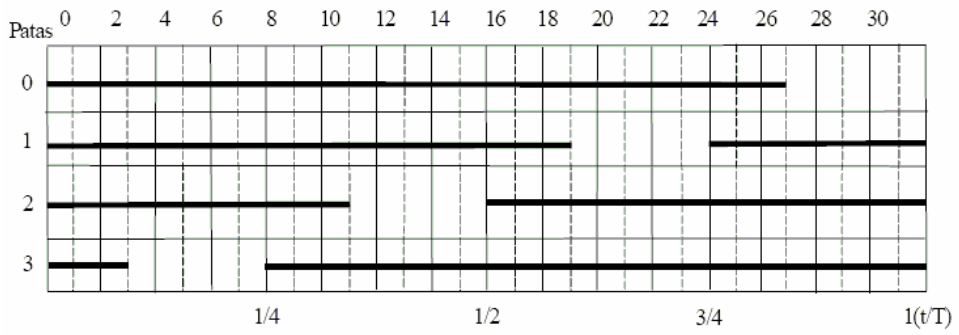


*Figura 4.14: Diagrama da andadura regular simétrica em onda periódica de ciclo completo.*

O ângulo de fase de uma pata *i* em relação à pata *0* é definido em número de acionamentos da pata *0* decorridos até o primeiro acionamento da pata *i.* Com esta estratégia de acionamento dos graus de liberdade, é possível variar o ângulo de fase e o fator de carga na montagem da andadura, e também realizar o acionamento diferencial das patas quando o robô estiver em uma trajetória curva em uma andadura qualquer [2].

Na andadura implementada para o robô Guará, três ou quatro patas sempre estarão apoiadas, o que permite que o robô caminhe em equilíbrio estaticamente estável.

O tempo total de cada fase é um múltiplo do número de colunas e o fator de carga é dado por $\beta = \frac{27}{32} = 0,84375$, onde *32* é o número total de colunas e *27* é o número de colunas correspondente ao período em que a pata está em empuxo.

Um *estágio* é a menor divisão do tempo total de ciclo *T* que permite definir com precisão a fase $\phi_i$. A andadura da Figura 4.14 tem 32 estágios, dos quais 27 em empuxo e 5 em vôo.

A Figura 4.15 mostra o robô em vista lateral, com trajetória da pata 2 quando se encontra em início de vôo.

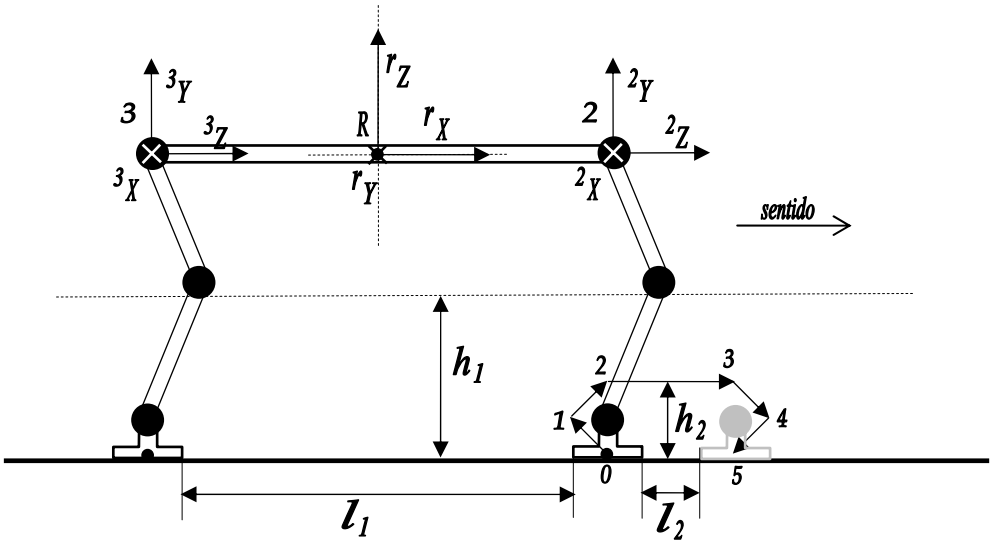


*Figura 4.15: Movimento da andadura de vôo da pata 2.*

Os pontos 0, 1, 2, 3, 4 e 5 são programados para dar a trajetória de vôo da pata. Esta trajetória se é poligonal, assemelhando-se ao movimento das pernas dos mamíferos. A altura $h_1$ representa a altura máxima que um obstáculo pode ter para ser transpassado debaixo da plataforma, entre as patas, enquanto que $l_1$ indica o comprimento entre as patas dianteiras e traseiras. A altura $h_2$ representa a altura máxima que um obstáculo pode ter para ser transpassado debaixo das patas enquanto que $l_2$ indica o comprimento máximo deste obstáculo.

**Movimento de deslocamento lateral**

O projeto do Guará levou em conta a necessidade de, além de implementar a andadura movimentando a perna em um plano vertical paralelo ao eixo longitudinal do robô, realizar um dos movimentos efetuados pelos mamíferos, que é o deslocamento lateral do centro de gravidade, e que resulta no movimento da perna também em um plano vertical perpendicular ao eixo longitudinal do robô [2].

No Guará, que caminha em equilíbrio estaticamente estável, este movimento é efetuado durante o período em que o robô permanece com as quatro patas apoiadas, no início da andadura e entre os eventos de vôo das patas, e posiciona a plataforma à esquerda, para vôo das patas *2* e *3*, e à direita, para vôo das patas *0* e *1*. A Figura 4.16 ilustra o movimento lateral do robô Guará. A largura $w_1$ representa a distância entre as patas e o ângulo $\theta_1$ a inclinação máxima que a perna do robô atinge lateralmente. A largura $w_2$ representa a largura máxima que o obstáculo pode ter para ser transpassado pela plataforma do robô. Logo, $w_2 = w_1 - 2.h_1.\tan\theta_1$.

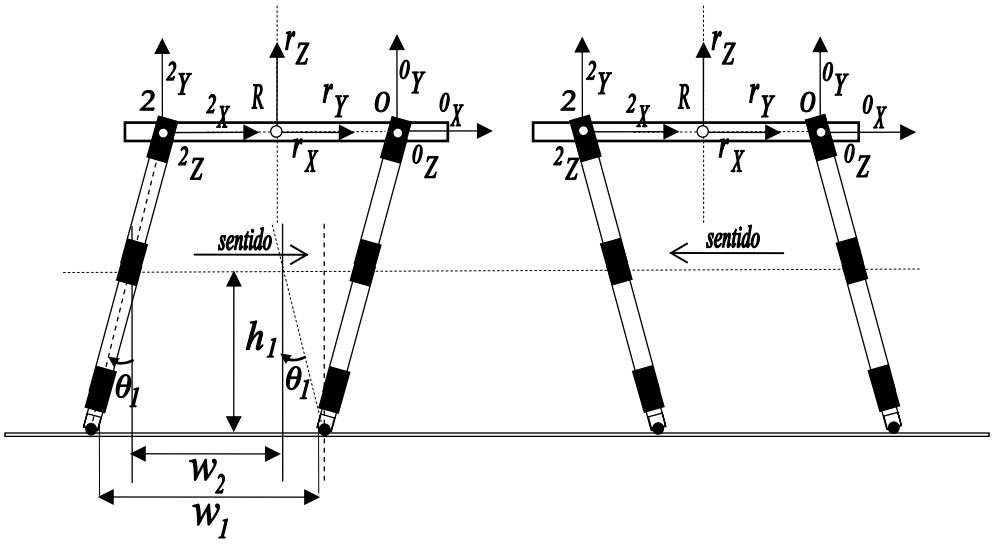


*Figura 4.16: Movimentos laterais da andadura do Guará.*

Para andar, o Guará combina movimentos de rolagem, mergulho e guinada, usando cinemática inversa. Os *movimentos retilíneos* são programados definindo-se um raio da curvatura $r_c$ acima de 30 m, enquanto que os *movimentos curvilíneos* são programados com raios cujos valores são definidos pela trajetória. A Figura 4.17 mostra, em vista superior, os parâmetros geométricos para a sua trajetória curvilínea.

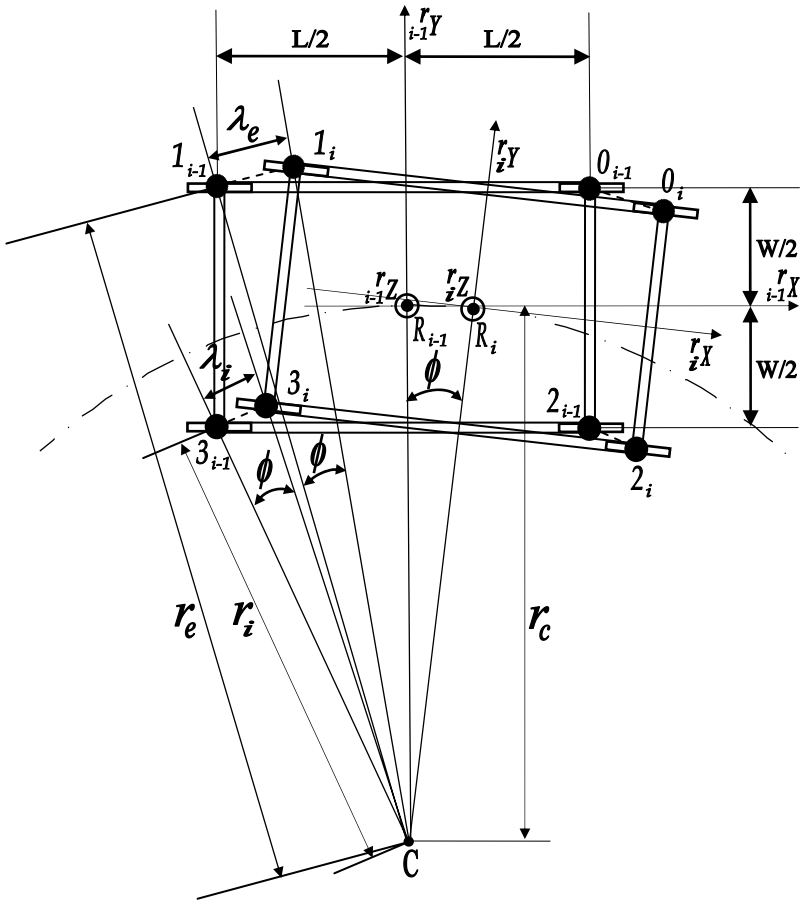


*Figura 4.17: Robô Guará em trajetória curvilínea.*

O Robô Guará realiza um incremento de arco $\phi$ em uma curva de raio $r_c$ sendo $r_e$ e $r_i$ os raios de curvatura externo e interno à curva, $\phi$ o incremento de arco para um acionamento, $\lambda_e$ e $\lambda_i$ as amplitudes do acionamento das pernas externa e interna. Por sua vez, *W* e *L* são a largura e o comprimento do robô, e os números identificam as patas nos instantes $i-1$ e $i$.

O percurso das patas externas e internas deve ser

$$\begin{aligned}\lambda_e &= \phi.r_e = \phi.\left[\left(r_c + \frac{W}{2}\right)^2 + \left(\frac{L}{2}\right)^2\right]^{1/2}, \\ \lambda_i &= \phi.r_i = \phi.\left[\left(r_c - \frac{W}{2}\right)^2 + \left(\frac{L}{2}\right)^2\right]^{1/2}.\end{aligned} \tag{4.80}$$

A amplitude do passo é sempre determinada para o percurso do centróide do robô. Para realizar uma curva de raio $r_c$ à direita, o robô corrige o comprimento do percurso apoiado das patas externas *0* e *1*, de acordo com a equação

$$\begin{aligned}{}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_0^i &= {}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_0^{i-1} + \begin{bmatrix}\lambda_e.\cos\phi\\-\lambda_e.\operatorname{sen}\phi\\0\end{bmatrix}, \\ {}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_1^i &= {}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_1^{i-1} + \begin{bmatrix}\lambda_e.\cos\phi\\\lambda_e.\operatorname{sen}\phi\\0\end{bmatrix},\end{aligned} \tag{4.81}$$

e para comando das patas internas *2* e *3*, de acordo com

$$\begin{aligned}{}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_2^i &= {}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_2^{i-1} + \begin{bmatrix}\lambda_i.\cos\phi\\-\lambda_i.\operatorname{sen}\phi\\0\end{bmatrix}, \\ {}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_3^i &= {}^r\begin{bmatrix}x\\y\\z\end{bmatrix}_3^{i-1} + \begin{bmatrix}\lambda_i.\cos\phi\\\lambda_i.\operatorname{sen}\phi\\0\end{bmatrix},\end{aligned} \tag{4.82}$$

onde ${}^r\begin{bmatrix}x & y & z\end{bmatrix}^{(i-1)T}$ e ${}^r\begin{bmatrix}x & y & z\end{bmatrix}^{(i)T}$ e são as coordenadas dos pontos de fixação das pernas à plataforma entre dois acionamentos $i-1$ e $i$.

## 4.9 Cubificação dos Obstáculos

A informação contida em uma imagem é representada por milhares de pontos, que precisam ser agrupados para reduzir o universo de dados. Técnicas como segmentação de imagem consistem em dividir a imagem em regiões que dizem respeito ao mesmo conteúdo e aplicação. Existem objetos de interesse em uma imagem e podem-se isolar aqueles *pixels* que fazem parte desses objetos. A segmentação deve parar quando os objetos de interesse numa aplicação forem isolados [1].

Tradicionalmente, a segmentação de imagem tem sido vista como um estágio prévio de processamento para identificação ou análise. Primeiro, a imagem é segmentada, e então processada. Uma aproximação geral, em muitos dos métodos de segmentação, é agrupar, de alguma forma, os *pixels* com mesma propriedade, isto é, mesma intensidade, cor ou região com textura semelhante.

No caso de imagens de profundidade, a segmentação é feita separando-se os *pixels* cuja intensidade $f(x, y) > 0$, ou seja, neste caso estes *pixels* representam os obstáculos. Durante o processo de mapeamento, os *pixels* dos obstáculos sofrem transformações geométricas e correção pelo algoritmo *ICP*, gerando pontos *3D* referenciados ao sistema de coordenadas do mundo.

Contudo há uma quantidade muito grande de pontos *3D* dos obstáculos, para viabilizar o processamento é feita uma amostragem estratégica destes pontos. Este processo é descrito como *Cubificação dos Obstáculos*.

Os pontos amostrados são agrupados para representarem pequenas áreas elementares. É definido um obstáculo elementar, que é uma fração do obstáculo, ocupando uma área elementar quadrada, sendo representado por apenas 3 pontos *3D*.

Seja $o_k$ um obstáculo elementar, onde $k = 1,2,3,...,n$ e $n$ indica a quantidade total de obstáculos elementares que compõem um obstáculo. O ponto $\mathbf{p_{1k}} = \begin{bmatrix} x_{1k} & y_{1k} & z_{1k} \end{bmatrix}^T$ representa o seu ponto superior esquerdo, $\mathbf{p_{2k}} = \begin{bmatrix} x_{2k} & y_{2k} & z_{2k} \end{bmatrix}^T$ representa o seu ponto inferior direito e $\mathbf{p_{ck}} = \begin{bmatrix} x_{ck} & y_{ck} & z_{ck} \end{bmatrix}^T$ representa o centro do quadrado, e cujas coordenadas *z* representam o maior valor de todos os pontos pertencentes a esta área elementar, ou seja, $z_{1k} = z_{2k} = z_{(\max)k} = \max(z_i), i = 1,2,3...m$ , onde *m* é a quantidade total de pontos contidos na área do obstáculo elementar $o_k$ .

Assim, para *m* pontos da nuvem de pontos, apenas três pontos são usados, possibilitando um mapeamento racional dos obstáculos e reduzindo consideravelmente o custo computacional dos algoritmos.

A cubificação dos obstáculos é dada em fração da área do quadro da imagem, sendo um parâmetro variável. Quanto maior o valor da fração, há menos pontos para processamento. Em contrapartida, a área e o volume calculados para o obstáculo serão mais imprecisos.

A Figura 4.18 mostra duas imagens. A Figura 4.18(1) contém a imagem de profundidade integrada com todos os pontos *3D*, e a Figura 4.18(2) contém a mesma imagem cubificada com áreas elementares de 20x20 $mm^2$ do quadro da imagem.

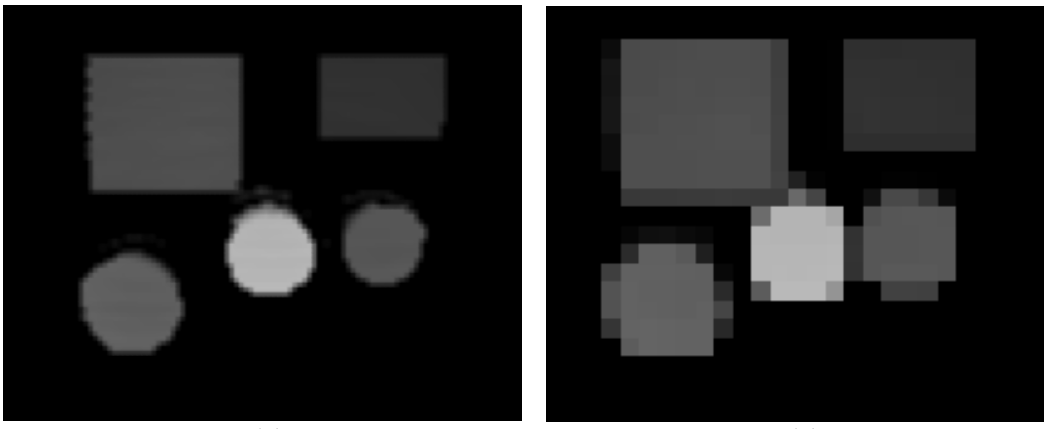


*Figura 4.18: (1) Imagem de profundidade integrada com todos os pontos 3D (2) Imagem integrada cubificada com pontos 3D representativos.*

## 4.10 Mapeamento da Superfície

Para mapear a superfície e os obstáculos, foi definido um conjunto de 20 distâncias e 25 pontos estratégicas, que ajuda na tomada de decisão de navegação do robô, conforme mostrado na Figura 4.19. Para garantir que o robô possa dar um passo, é necessário verificar se há obstáculos na direção do movimento, sob a plataforma do robô, nas laterais e entre as patas.

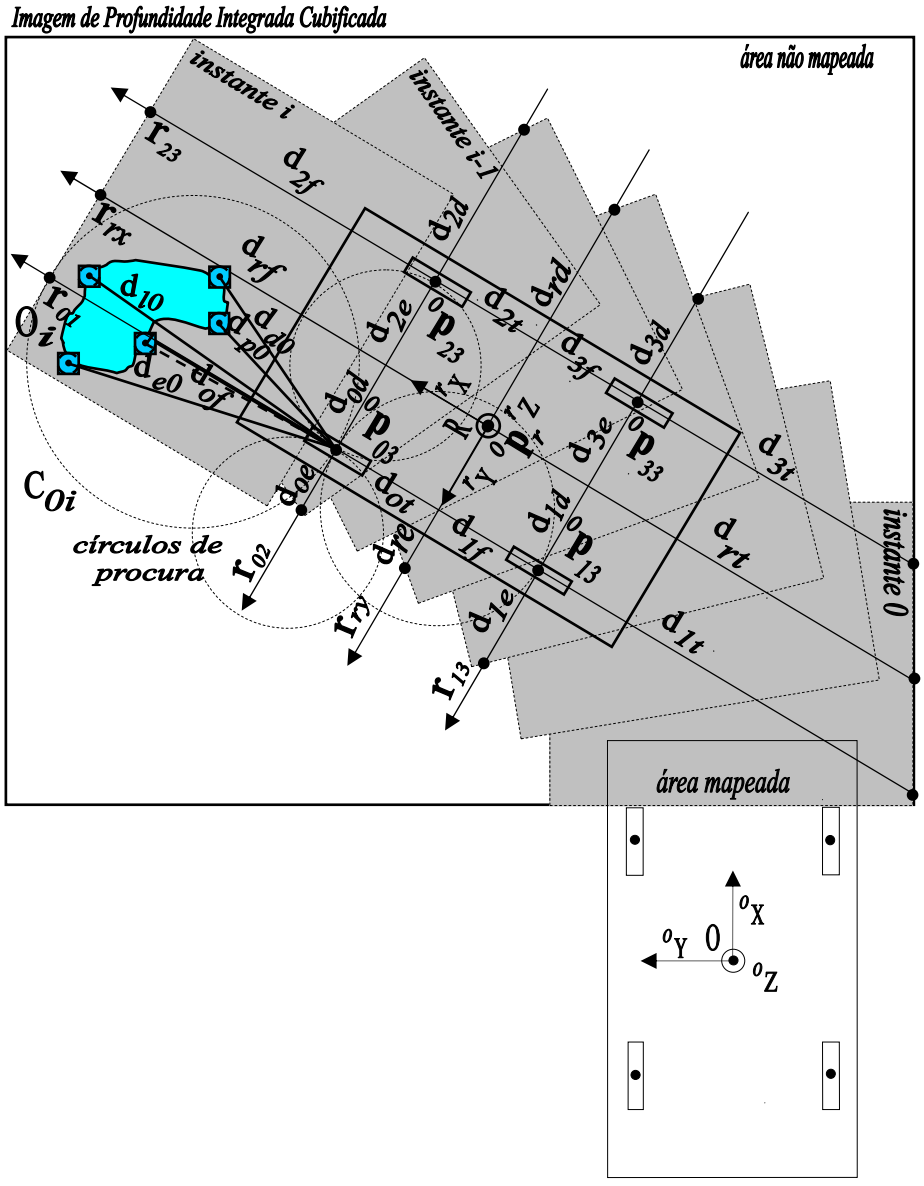


*Figura 4.19: Distâncias estratégicas entre os pés das patas e os obstáculos.*

No instante *i* de análise, o vetor ${}^{o}\mathbf{p}_r = \begin{bmatrix} {}^{o}x_r & {}^{o}y_r & {}^{o}z_r \end{bmatrix}^T$ indica a posição do referencial do robô, ${}^{o}\mathbf{p}_{03} = \begin{bmatrix} {}^{o}x_{03} & {}^{o}y_{03} & {}^{o}z_{03} \end{bmatrix}^T$ a posição da pata 0, ${}^{o}\mathbf{p}_{13} = \begin{bmatrix} {}^{o}x_{13} & {}^{o}y_{13} & {}^{o}z_{13} \end{bmatrix}^T$ a posição da pata 1, ${}^{o}\mathbf{p}_{23} = \begin{bmatrix} {}^{o}x_{23} & {}^{o}y_{23} & {}^{o}z_{23} \end{bmatrix}^T$ a posição da pata 2 e ${}^{o}\mathbf{p}_{33} = \begin{bmatrix} {}^{o}x_{33} & {}^{o}y_{33} & {}^{o}z_{33} \end{bmatrix}^T$ a posição da pata 3, todos em relação ao referencial do mundo. Através destes vetores são geradas seis equações de reta $r_{01}$, $r_{02}$, $r_{13}$ , $r_{23}$, $r_{rx}$ e $r_{ry}$ com os coeficientes angulares

$$m_{01} = \frac{{}^{o}x_{03} - {}^{o}x_{13}}{{}^{o}y_{03} - {}^{o}y_{13}}, \qquad m_{02} = \frac{{}^{o}x_{03} - {}^{o}x_{23}}{{}^{o}y_{03} - {}^{o}y_{23}}, \qquad m_{13} = \frac{{}^{o}x_{13} - {}^{o}x_{33}}{{}^{o}y_{13} - {}^{o}y_{33}}, \qquad m_{23} = \frac{{}^{o}x_{23} - {}^{o}x_{23}}{{}^{o}y_{23} - {}^{o}y_{23}},$$

$$m_{rx} = \frac{({}^{o}x_{03} + {}^{o}x_{23})/2 - ({}^{o}x_{13} + {}^{o}x_{33})/2}{({}^{o}y_{03} + {}^{o}y_{23})/2 - ({}^{o}y_{13} + {}^{o}y_{33})/2} \quad \text{e} \quad m_{ry} = \frac{({}^{o}x_{03} + {}^{o}x_{13})/2 - ({}^{o}x_{23} + {}^{o}x_{33})/2}{({}^{o}y_{03} + {}^{o}y_{13})/2 - ({}^{o}y_{23} + {}^{o}y_{33})/2},$$

respectivamente. Todos os coeficientes angulares são calculados levando em consideração os casos particulares dos pontos de singularidade, quando o ângulo de declividade das retas é $\pm 90^0$. Tais equações são

$$\begin{aligned}
&r_{01} : x - {}^{o}x_{03} = m_{01}.(y - {}^{o}y_{03}),\\
&r_{02} : x - {}^{o}x_{03} = m_{02}.(y - {}^{o}y_{03}),\\
&r_{13} : x - {}^{o}x_{13} = m_{13}.(y - {}^{o}y_{13}),\\
&r_{23} : x - {}^{o}x_{23} = m_{23}.(y - {}^{o}y_{23}),\\
&r_{rx} : x - ({}^{o}x_{03} + {}^{o}x_{23})/2 = m_{rx}.(y - ({}^{o}y_{03} + {}^{o}y_{23})/2),\\
&r_{ry} : x - ({}^{o}x_{03} + {}^{o}x_{13})/2 = m_{ry}.(y - ({}^{o}y_{03} + {}^{o}y_{13})/2),
\end{aligned} \tag{4.83}$$

onde $(x, y)$ representa um ponto qualquer pertencente à reta em questão.

As distâncias são $d_{if}$ à frente, $d_{it}$ atrás, $d_{ie}$ à esquerda e $d_{id}$ à direita dos pontos, onde $i = 0,1,2,3,r$ representa o número de cada perna e *r* do referencial do robô. As distâncias são calculadas do ponto até encontrar um obstáculo mais próximo, nas direções definidas pelas retas.

**Exemplificando para a distância $d_{0f}$**

Na Figura 4.19, seja o obstáculo cubificado $O_i$, formado por $n$ obstáculos elementares $o_k$, tal que $O_i = \sum_{k=1}^{n} o_k$, contidos dentro do círculo de procura frontal da pata 0 $C_{0i}$.

Os obstáculos elementares são a base de cálculo para todas as distâncias e pontos estratégicos do mapeamento. Seja um obstáculo elementar $o_k$, usado para o cálculo da distância frontal $d_{0f}$ da pata 0 até o obstáculo mais próximo. Há 3 pontos que definem o obstáculo $o_k$: o ponto superior esquerdo ${}^{o}\mathbf{p}_{1k} = \begin{bmatrix} {}^{o}x_{1k} & {}^{o}y_{1k} & {}^{o}z_{1k} \end{bmatrix}^T$, o ponto inferior direito ${}^{o}\mathbf{p}_{2k} = \begin{bmatrix} {}^{o}x_{2k} & {}^{o}y_{2k} & {}^{o}z_{2k} \end{bmatrix}^T$ e o ponto do centro ${}^{o}\mathbf{p}_{ck} = \begin{bmatrix} {}^{o}x_{ck} & {}^{o}y_{ck} & {}^{o}z_{ck} \end{bmatrix}^T$. Seja $r_k$ o raio de um círculo inscrito neste obstáculo. Para saber se este obstáculo se encontra na direção definida pela reta $r_{01}$, é calculada da distância do centro do obstáculo até a reta $r_{01}$ e caso esta distância for inferior ao raio $r_k$ do círculo inscrito do obstáculo, conclui-se que este obstáculo corta a reta $r_{01}$.

A distância do ponto do centro do obstáculo, normal à reta $r_{01} : a_{01}.x + b_{01}.y + c_{01}$ é dada por

$$d({}^{o}\mathbf{p}_{ck}, r_{01}) = \left| \frac{a_{01}.{}^{o}x_{ck} + b_{01}.{}^{o}y_{ck} + c_{01}}{\sqrt{a_{01}^2 + b_{01}^2}} \right|, \tag{4.84}$$

onde $a_{01} = 1$, $b_{01} = -m_{01}$ e $c_{01} = m_{01}.{}^{o}y_{03} - {}^{o}x_{03}$. Neste caso, a distância $d_{0f}$ é aproximada pela distância da pata 0 até o centro do obstáculo $o_k$ pois o raio $r_c$ é pequeno, sendo desprezível perante o valor da distância, reduzindo assim o esforço computacional, logo

$$d_{0f} \simeq d({}^{o}\mathbf{p}_{ck}, {}^{o}\mathbf{p}_{03}) = \sqrt{({}^{o}x_{ck} - {}^{o}x_{03})^2 + ({}^{o}y_{ck} - {}^{o}y_{03})^2}. \tag{4.85}$$

Caso haja outros obstáculos na direção da reta $r_{01}$, o procedimento é repetido, e, posteriormente, as várias distâncias $d_{0f}$ são comparadas e é escolhida a menor distância, ou seja, o obstáculo mais próximo é encontrado.

Da mesma maneira, as outras distâncias são calculadas, gerando assim um conjunto de dados relevantes para a tomada de decisão na navegação.

Dentro do círculo de procura $C_{0i}$, são obtidas 4 distâncias $d_{e0}, d_{d0}$, $d_{p0}$ e $d_{l0}$ associadas aos pontos de centro ${}^{o}\mathbf{p}_{e0} = \begin{bmatrix} {}^{o}x_{e0} & {}^{o}y_{e0} & {}^{o}z_{e0} \end{bmatrix}^T$, ${}^{o}\mathbf{p}_{d0} = \begin{bmatrix} {}^{o}x_{d0} & {}^{o}y_{d0} & {}^{o}z_{d0} \end{bmatrix}^T$, ${}^{o}\mathbf{p}_{p0} = \begin{bmatrix} {}^{o}x_{p0} & {}^{o}y_{p0} & {}^{o}z_{p0} \end{bmatrix}^T$ e ${}^{o}\mathbf{p}_{l0} = \begin{bmatrix} {}^{o}x_{l0} & {}^{o}y_{l0} & {}^{o}z_{l0} \end{bmatrix}^T$, respectivamente. O ponto ${}^{o}\mathbf{p}_{e0}$ indica o centro do obstáculo elementar mais próximo à esquerda da pata 0, o ponto ${}^{o}\mathbf{p}_{d0}$ indica o centro do obstáculo elementar mais próximo à direita da pata 0, o ponto ${}^{o}\mathbf{p}_{p0}$ indica o centro do obstáculo elementar mais próximo da pata 0 e o ponto ${}^{o}\mathbf{p}_{l0}$ indica o centro do obstáculo elementar mais distante da pata 0.

Para achar o ponto da esquerda, é escolhido um ponto do centro do lado esquerdo cuja distância perpendicular até a reta $r_{01}$ é a maior. Para achar o ponto da direita, é escolhido um ponto do centro do lado direito cuja distância perpendicular até a reta $r_{01}$ é a maior.

Para achar o ponto mais próximo, o ponto do centro que tenha a menor distância até a pata 0 é escolhido. Para achar o ponto mais distante, o ponto do centro que tenha a maior distância até a pata 0 é escolhido.

A Figura 4.20 mostra todos os círculos de procura usados para identificação e mapeamento de obstáculos. Um obstáculo já cubificado é encontrado e mapeado no canto superior esquerdo pertencente aos círculos de procura das patas 0 e 2. Os seguintes pontos foram obtidos em relação à pata 0: ${}^{o}\mathbf{p}_{e0} = \begin{bmatrix} 543 & 568 & 9 \end{bmatrix}^T$, ${}^{o}\mathbf{p}_{d0} = \begin{bmatrix} 643 & 368 & 60 \end{bmatrix}^T$, ${}^{o}\mathbf{p}_{p0} = \begin{bmatrix} 503 & 408 & 15 \end{bmatrix}^T$, ${}^{o}\mathbf{p}_{l0} = \begin{bmatrix} 703 & 508 & 31 \end{bmatrix}^T$ e a distância frontal $d_{0f} = 323$, sendo todas as unidades em *mm*.

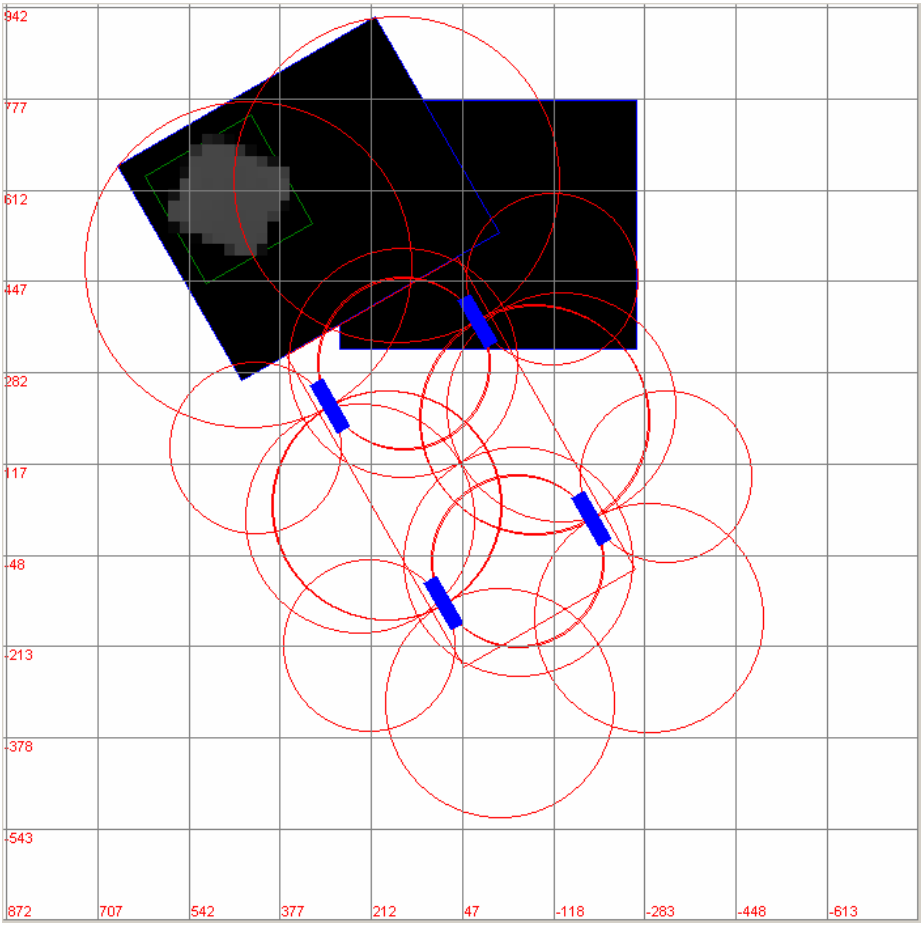


*Figura 4.20: Círculos de procura para detecção de obstáculos (em mm) .*

À medida que o robô navega pela superfície, os obstáculos vão sendo mapeados e guardados em uma lista com seus descritores, pois eventualmente o robô pode retornar ao mesmo lugar e novos dados do obstáculo são obtidos e integrados aos dados já existentes de uma outra posição. Esta característica permite um refinamento dos dados do obstáculo, acrescentando dados de zonas de sombra, por exemplo, que podem não ter sido mapeados na primeira varredura sobre o obstáculo.

A Figura 4.21 mostra uma lista de obstáculos, que, após o mapeamento feito pelo robô, foi armazenada em arquivo. Cada obstáculo tem um identificador único e uma série de descritores. Os obstáculos cubificados são mostrados em tons de cinza, contudo eles têm uma cor representativa associada ao seu identificador, conforme mostrado na Figura 4.22.

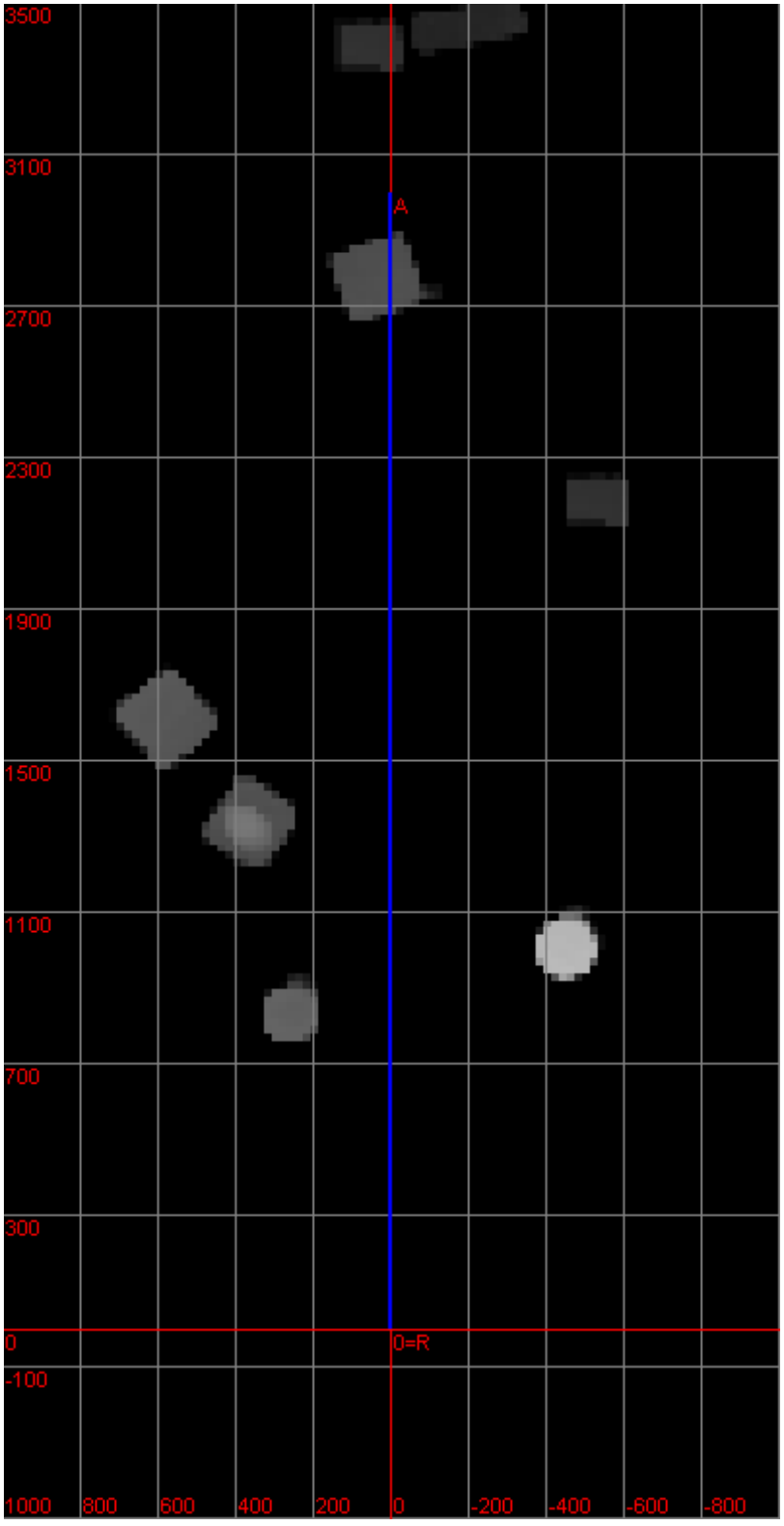


*Figura 4.21: Mapeamento dos obstáculos da superfície (em mm).*

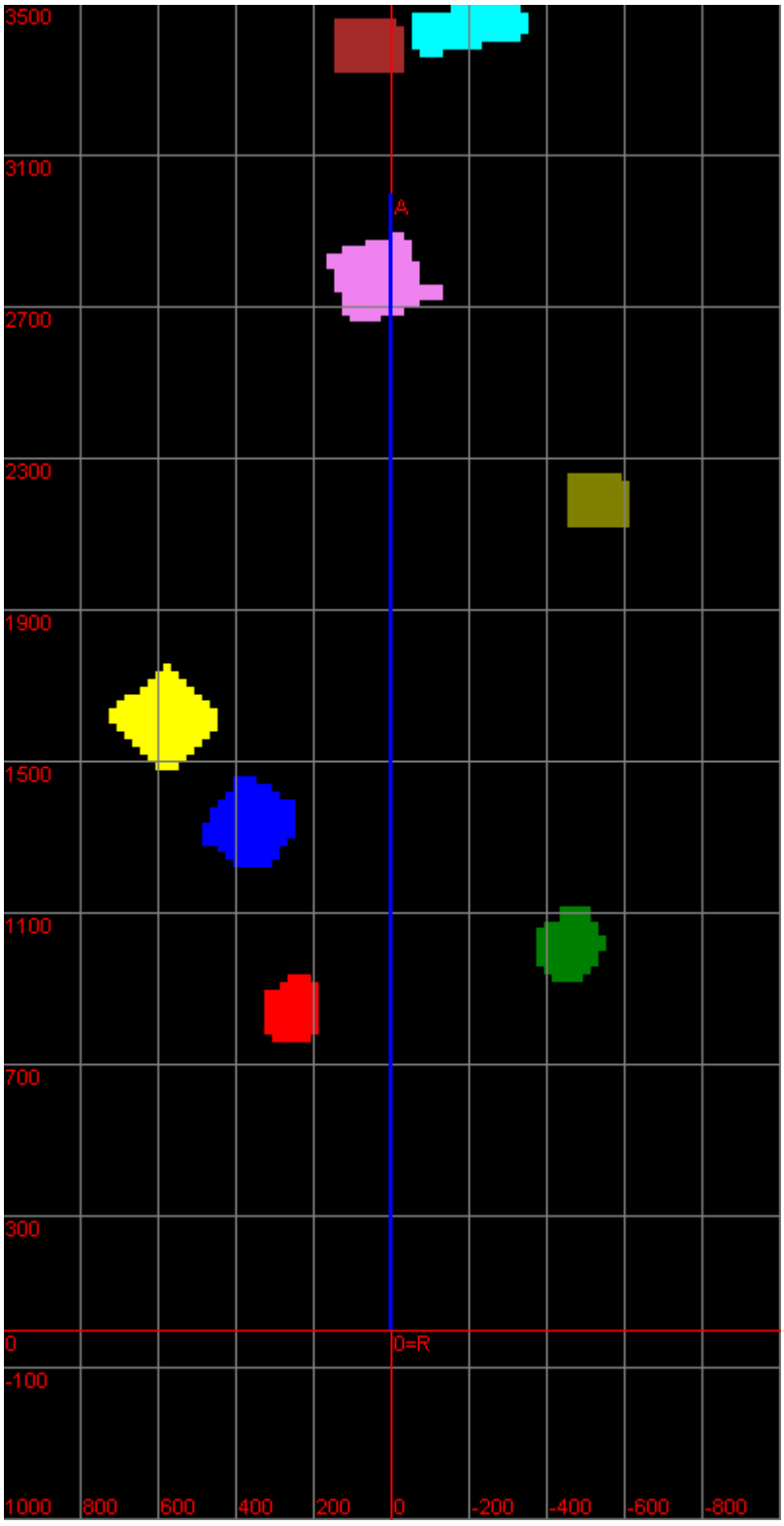


*Figura 4.22: Identificação dos obstáculos na superfície (em mm).*

## 4.11 Descritores dos Obstáculos

No desenvolvimento de uma aplicação de visão para robótica há sempre uma pergunta chave: quais características das imagens seriam mais informativas sobre o problema em questão?

Um descritor pode representar um valor, uma característica, um conjunto de dados reduzidos de um obstáculo, uma função matemática originalmente obtida de uma imagem. Desta forma, a imagem pode ser substituída por uma série de descritores, devidamente escolhidos para bem representá-la, com a vantagem de que o armazenamento utiliza menos memória.

A vantagem dos descritores é o uso dos seus dados em técnicas de controle e tomada de decisão. O fato de usar imagens de profundidade para obtenção de descritores torna o processo mais fácil. Além do mais, as imagens já foram pré-processadas usando a técnica de cubificação, e os obstáculos já isolados e identificados.

Do processo de cubificação, seja o obstáculo elementar $o_k$ representado por 3 pontos: $\mathbf{p_{1k}} = \begin{bmatrix} x_{1k} & y_{1k} & z_{1k} \end{bmatrix}^T$, ponto superior esquerdo, $\mathbf{p_{2k}} = \begin{bmatrix} x_{2k} & y_{2k} & z_{2k} \end{bmatrix}^T$, ponto inferior direito e $\mathbf{p_{ck}} = \begin{bmatrix} x_{ck} & y_{ck} & z_{ck} \end{bmatrix}^T$, o centro do obstáculo elementar. Um obstáculo $O_i$ é representado por um grupo de obstáculos elementares vizinhos, tal que

$$O_i = \sum_{k=1}^{n} o_k . \tag{4.86}$$

Este obstáculo $O_i$ representa uma estrutura de dados com custo computacional baixo, possibilitando a extração de vários descritores, necessários no controle de andadura do robô, definidos a seguir.

### 4.11.1 Área do Obstáculo

A área total $A_i$ do obstáculo $O_i$ é a soma total das áreas dos obstáculos elementares, como segue

$$A_i = \sum_{k=1}^{n} (x_{2k} - x_{1k}).(y_{2k} - y_{1k}). \tag{4.87}$$

### 4.11.2 Volume do Obstáculo

O volume total $V_i$ do obstáculo $O_i$ é a soma dos produtos das alturas dos centros dos obstáculos elementares pelas suas áreas, ou seja,

$$V_i = \sum_{k=1}^{n} z_{ck}.(x_{2k} - x_{1k}).(y_{2k} - y_{1k}). \tag{4.88}$$

### 4.11.3 Centróide do Obstáculo

O centróide do obstáculo representa o ponto designado pela média ponderada pelo volume elementar $v_k$ das coordenadas de todos os pontos 3D que formam o obstáculo. Seja o volume elementar $v_k = z_{ck}.(x_{2k} - x_{1k}).(y_{2k} - y_{1k})$, logo

$$\mathbf{p}_{ci} = \frac{1}{V_i}\left[\sum_{k=1}^{n} v_k.x_{ck} \quad \sum_{k=1}^{n} v_k.y_{ck} \quad \sum_{k=1}^{n} v_k.z_{ck}\right]^t. \tag{4.89}$$

### 4.11.4 Dimensões do Obstáculo

Os descritores largura, altura e comprimento definem a dimensão de um paralelepípedo que circunscreve o obstáculo. Estes descritores são necessários para usar na técnica de controle do próximo capítulo.

Os descritores largura e comprimento dependem da orientação de posição do robô, portanto eles são calculados no momento em que o robô define a sua orientação. O descritor altura representa a altura máxima do obstáculo. A Figura 4.23 mostra estes descritores.

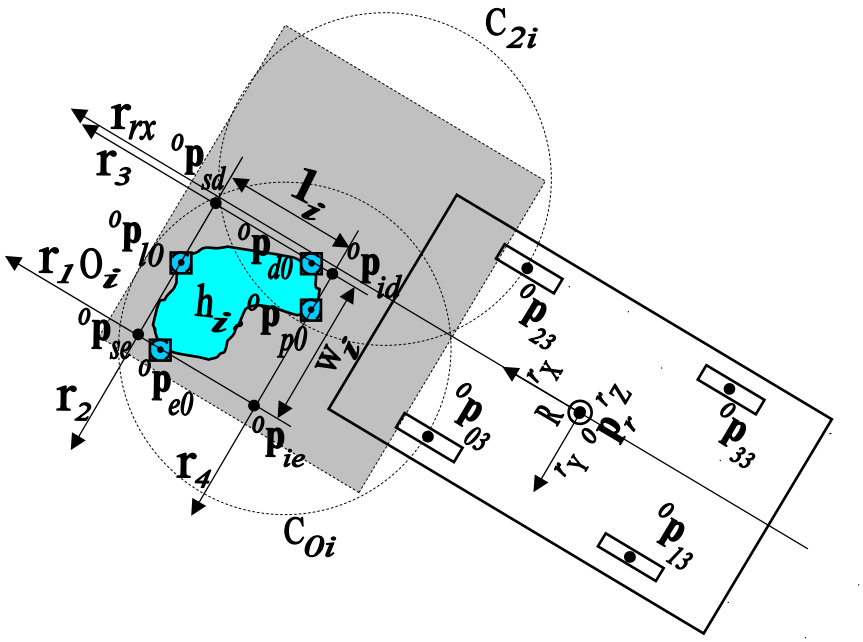


*Figura 4.23: Dimensões do obstáculo.*

Os quatros obstáculos elementares da Figura 4.23 representam o ponto mais próximo ${}^{o}\mathbf{p}_{p0} = \begin{bmatrix} {}^{o}x_{p0} & {}^{o}y_{p0} & {}^{o}z_{p0} \end{bmatrix}^{T}$, o ponto mais distante ${}^{o}\mathbf{p}_{l0} = \begin{bmatrix} {}^{o}x_{l0} & {}^{o}y_{l0} & {}^{o}z_{l0} \end{bmatrix}^{T}$, o ponto mais à esquerda ${}^{o}\mathbf{p}_{e0} = \begin{bmatrix} {}^{o}x_{e0} & {}^{o}y_{e0} & {}^{o}z_{e0} \end{bmatrix}^{T}$, e o ponto mais à direita ${}^{o}\mathbf{p}_{d0} = \begin{bmatrix} {}^{o}x_{d0} & {}^{o}y_{d0} & {}^{o}z_{d0} \end{bmatrix}^{T}$, obtidos do mapeamento do obstáculo, todos em relação à pata 0.

Sejam ${}^{o}\mathbf{p}_{se} = \begin{bmatrix} {}^{o}x_{se} & {}^{o}y_{se} & {}^{o}z_{se} \end{bmatrix}^{T}$, definido como o ponto superior esquerdo do obstáculo, ${}^{o}\mathbf{p}_{sd} = \begin{bmatrix} {}^{o}x_{sd} & {}^{o}y_{sd} & {}^{o}z_{sd} \end{bmatrix}^{T}$, o ponto superior direito, ${}^{o}\mathbf{p}_{ie} = \begin{bmatrix} {}^{o}x_{ie} & {}^{o}y_{ie} & {}^{o}z_{ie} \end{bmatrix}^{T}$, o ponto inferior esquerdo, e ${}^{o}\mathbf{p}_{id} = \begin{bmatrix} {}^{o}x_{id} & {}^{o}y_{id} & {}^{o}z_{id} \end{bmatrix}^{T}$, o ponto inferior direito.

A orientação do robô é definida pela reta $r_{rx}$ da equação (4.83), cujo coeficiente angular é $m_{rx}$. Portanto para achar os pontos ${}^{o}\mathbf{p}_{se}$, ${}^{o}\mathbf{p}_{sd}$, ${}^{o}\mathbf{p}_{ie}$ e ${}^{o}\mathbf{p}_{id}$, é necessário gerar as equações de 4 retas, no plano $z = 0$, como segue

$$
\begin{aligned}
r_1 &: x - {}^{o}x_{e0} = m_{rx}.(y - {}^{o}y_{03}),\\
r_2 &: x - {}^{o}x_{l0} = -\frac{(y - {}^{o}y_{lo})}{m_{rx}},\\
r_3 &: x - {}^{o}x_{d0} = m_{rx}.(y - {}^{o}y_{d0}),\\
r_4 &: x - {}^{o}x_{p0} = -\frac{(y - {}^{o}y_{po})}{m_{rx}},
\end{aligned}
\tag{4.90}
$$

e resolver os sistemas de equações para encontrar os pontos de interseção das retas. Portanto, as coordenadas de ${}^{o}\mathbf{p}_{se}$ representam a interseção das retas $r_1$ com $r_2$, as coordenadas de ${}^{o}\mathbf{p}_{sd}$ representam a interseção das retas $r_2$ com $r_3$, as coordenadas de ${}^{o}\mathbf{p}_{id}$ representam a interseção das retas $r_3$ com $r_4$ e as coordenadas de ${}^{o}\mathbf{p}_{ie}$ representam a interseção das retas $r_1$ com $r_4$. Os casos de singularidade da reta $r_{rx}$ são tratados como casos particulares.

A altura do obstáculo é representada pela maior coordenada *z* entre as coordenadas *z* dos obstáculos elementares, ou seja,

$$h_i = \max(z_k), k = 1, 2, 3...n. \tag{4.91}$$

A largura $w_i$ é a distância euclidiana entre os pontos ${}^{o}\mathbf{p}_{se}$ e ${}^{o}\mathbf{p}_{sd}$, ou entre os pontos ${}^{o}\mathbf{p}_{ie}$ e ${}^{o}\mathbf{p}_{id}$, dada por

$$w_i = d({}^{o}\mathbf{p}_{se}, {}^{o}\mathbf{p}_{sd}) = \sqrt{({}^{o}x_{se} - {}^{o}x_{sd})^2 + ({}^{o}y_{se} - {}^{o}y_{sd})^2}. \tag{4.92}$$

O comprimento $l_i$ é a distância euclidiana entre os pontos ${}^{o}\mathbf{p}_{se}$ e ${}^{o}\mathbf{p}_{ie}$ ou entre os pontos ${}^{o}\mathbf{p}_{sd}$ e ${}^{o}\mathbf{p}_{id}$, dada por

$$l_i = d({}^{o}\mathbf{p}_{se}, {}^{o}\mathbf{p}_{ie}) = \sqrt{({}^{o}x_{se} - {}^{o}x_{ie})^2 + ({}^{o}y_{se} - {}^{o}y_{ie})^2}. \tag{4.93}$$

Para o obstáculo da Figura 4.20 , os seguintes descritores foram obtidos, levando em consideração a orientação da plataforma do robô:

- ✓ identificador: 1
- ✓ volume: 433.360 $mm^3$
- ✓ área: 7.200 $mm^2$
- ✓ centróide: ${}^{o}\mathbf{p}_{c} = \begin{bmatrix} 598 & 467 & 66 \end{bmatrix}^{T}$ em mm
- ✓ pontos superiores: ${}^{o}\mathbf{p}_{se} = \begin{bmatrix} 636 & 622 & 0 \end{bmatrix}^{T}$, ${}^{o}\mathbf{p}_{sd} = \begin{bmatrix} 747 & 430 & 0 \end{bmatrix}^{T}$ em mm
- ✓ pontos inferiores: ${}^{o}\mathbf{p}_{ie} = \begin{bmatrix} 442 & 510 & 0 \end{bmatrix}^{T}$, ${}^{o}\mathbf{p}_{id} = \begin{bmatrix} 552 & 318 & 0 \end{bmatrix}^{T}$ em mm
- ✓ altura: 71 mm
- ✓ largura: 221 mm
- ✓ comprimento: 224 mm

Os descritores definidos neste capítulo são dados fundamentais para a estratégia de controle descrita no próximo capítulo. Houve a preocupação de se gerar descritores de fácil processamento, representando bem as imagens de profundidade, de onde eles foram extraídos.

A geração dos descritores se tornou um processo simplificado graças ao fato de se usar imagens de profundidade. No estágio da segmentação dos obstáculos, a técnica da cubificação reduziu substancialmente a quantidade de pontos de análise, viabilizando o rápido processamento dos descritores.

# Capítulo 5 – Controle de Navegação do Robô Guará

## 5.1 Introdução

Este capítulo detalha a estratégia de controle proposta para as tomadas de decisão do robô durante a navegação. O método proposto usa *Navegação Reativa* com um sistema baseado em Árvore de Decisão Binária, com regras de percepção e ação que permitem ao robô transpassar ou desviar-se de obstáculos, enquanto se locomove de forma otimizada em direção ao ponto de destino.

O objetivo deste capítulo é mostrar uma aplicação que utiliza os dados de mapeamento da superfície para robôs quadrúpedes. Devido à grande quantidade de situações de posicionamento de obstáculos em relação ao robô, a implementação desta técnica de controle se restringiu à análise da influência de um obstáculo, propondo algoritmos para 3 situações:

- ✓ robô encontra um obstáculo e passa com a plataforma sobre ele;
- ✓ robô encontra um obstáculo e passa com a pata sobre ele;
- ✓ robô encontra um obstáculo e desvia-se pela esquerda ou pela direita.

O capítulo aborda a comunicação entre o sistema de controle do robô e o sistema de visão, a arquitetura de controle implementada, com os seus requisitos necessários de condições de estabilidade e limites cinemáticos do robô e também dos obstáculos. A árvore binária de decisão com os critérios e ações implementados é discriminada.

## 5.2 Comunicação com o Sistema de Controle do Robô Guará

O robô Guará tem o seu sistema de controle independente rodando em aplicação de 32 *bits* para ambiente *Windows*. O sistema de visão tem também uma aplicação de 32 *bits*, desenvolvida para a plataforma *.Net* no sistema operacional *Windows*. Ambos os sistemas trocam mensagens através de comunicação entre processos, usando *Sockets*.

Para cada passo executado do robô, os seguintes dados são recebidos do sistema de controle do robô:

a) *byte* identificador do pacote;
b) número do passo dado pelo robô;
c) vetor de translação do centro geométrico do robô em relação ao referencial do mundo;
d) matriz de rotação do centro geométrico do robô em relação ao referencial do mundo;
e) vetor de coordenadas da pata 0 junta 3 em relação ao referencial do robô;
f) vetor de coordenadas da pata 1 junta 3 em relação ao referencial do robô;
g) vetor de coordenadas da pata 2 junta 3 em relação ao referencial do robô;
h) vetor de coordenadas da pata 3 junta 3 em relação ao referencial do robô.

Ao receber o pacote completo, o sistema de visão aciona o sistema do laser para fazer uma varredura e obter a imagem de profundidade do quadro, que é integrada à imagem de profundidade anterior.

A estratégia de controle proposta toma decisão de navegação em função dos obstáculos mapeados, e os seguintes dados são enviados para o sistema de controle do robô:

a) *byte* identificador do pacote;
b) matriz de correção da rotação da plataforma do robô em relação ao referencial do mundo, caso o algoritmo *ICP* tenha convergido, ou a matriz identidade, caso o *ICP* tenha falhado;
c) vetor de correção da translação do centro geométrico do robô em relação ao referencial do mundo, caso o algoritmo *ICP* tenha convergido, ou vetor de translação do centro geométrico original do robô, caso o *ICP* tenha falhado;
d) *byte* da ação de tomada de decisão do sistema de controle;
e) raio de curvatura a ser dado pelo robô.

## 5.3 Arquitetura de Controle para Navegação Reativa do Robô Guará

As técnicas de navegação de robôs móveis vêm sendo estudadas por vários pesquisadores. Estas técnicas apresentam vantagens e desvantagens e variam em função do tipo de robô e sua tarefa, e da dinâmica, estrutura e característica do ambiente de navegação.

O problema de navegação de robôs móveis é grande e complexo. A superfície que o robô pode navegar varia de ambiente interno estático, com obstáculos em posições determinadas, até ambientes externos dinâmicos, com obstáculos indeterminados que mudam de posição. O ambiente pode ser estruturado, semi-estruturado e não estruturado.

Em [49] é feita uma pesquisa das principais técnicas de navegação de robôs móveis. Os robôs móveis são enquadrados em duas categorias: holonômicos e não-holonômicos. Robôs holonômicos são capazes de variar cada componente de sua posição e sua orientação de forma independente, enquanto que os robôs não-holonômicos se locomovem apenas em um número limitado de direções, em que mudança na sua orientação causa mudança na sua posição.

Na tarefa do robô, um caminho pré-determinado é definido, contendo um alvo-destino. O robô é programado para seguir uma trajetória até o alvo. Neste caso, as técnicas de navegação de robôs podem ser do tipo *global*, baseadas em informações de mapas pré-definidos, sendo uma navegação planejada a priori, e do tipo *local*, baseadas em informação sensorial, sendo uma navegação reativa.

No geral, os métodos de navegação planejada não são aplicáveis se o ambiente é dinâmico e com comportamento desconhecido ou gradualmente descoberto. Além do mais, quando a modelagem do ambiente e a odometria do robô têm incertezas, devido às imprecisões das medidas, a execução de uma trajetória geométrica teórica não é realista, e o robô corre o risco de colidir com obstáculos [50].

A Navegação Reativa, escolhida para o Guará, é, então, um método robusto para lidar com o problema de navegação do robô, porque leva em consideração a realidade das condições do ambiente. Este método é baseado em um processo de *percepção-e-ação*, continuamente executado. Inicialmente a informação sensorial é obtida; então este método processa o melhor comando de navegação de modo a evitar colisões com obstáculos e conduzir o robô ao seu alvo de destino.

A Navegação Reativa é muito usada em robôs móveis a rodas. Para os robôs quadrúpedes, há estratégias similares. Em [51] a técnica de *campo potencial repulsivo* é mostrada, a qual usa um rastreador a laser para descobrir a menor distância até os obstáculos. Em [52] a detecção de obstáculos é feita com uma câmera e sensores ultra-sônicos, os obstáculos são detectados apenas na frente do robô, e são considerados paralelepípedos retangulares.

A Navegação Reativa, embora viabilize que o robô ande em ambientes dinâmicos e desconhecidos, nem sempre traz soluções ótimas, e corre o risco de deixar o robô em situações de armadilha (*trap*), porque ela só usa uma fração local da informação disponível.

Este paradigma de navegação é baseado em definir o conjunto de situações que descrevem os estados reativos das entidades e definir ações associadas com cada situação.

O algoritmo proposto como técnica para navegação reativa do robô Guará baseia-se em satisfazer 3 condições para o funcionamento, discriminadas a seguir:

### 5.3.1 Condição de Estabilidade

Esta condição satisfaz o equilíbrio da plataforma do robô. Os movimentos principais são precedidos de movimentos secundários axiais e transversais da plataforma, planejados para permitir um posicionamento que sempre maximize a margem de estabilidade do robô.

O movimento de deslocamento lateral do Guará resulta no aumento da margem de estabilidade do robô, que é representada pela menor distância da projeção vertical do centro

geométrico aos lados do polígono de apoio, pois o movimento lateral move o centro de gravidade do robô para dentro deste polígono, conforme ilustrado na Figura 5.1.

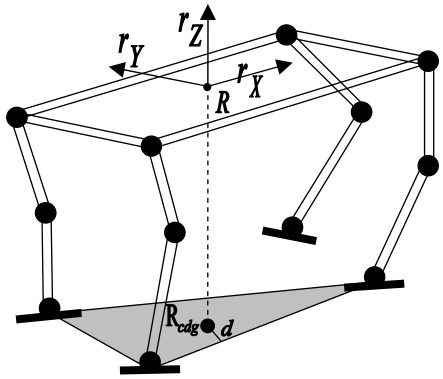


*Figura 5.1: Robô caminhando em condição de equilíbrio estático.*

Para navegação em curvas, o algoritmo incremental através de segmentos de reta no espaço de coordenadas, que permite variar o raio e o ângulo de curva com o robô em movimento, sincroniza uma seqüência padronizada de movimentos da andadura e preserva o equilíbrio e estabilidade do robô [2].

### 5.3.2 Condição de Cinemática

O robô Guará pode ser interpretado como sendo formado por quatro robôs manipuladores com 4 graus de liberdade cada, fixados em uma plataforma móvel. Cada perna, portanto, funciona como um manipulador de quatro graus de liberdade.

O robô foi programado para andar em trajetória retilínea ou curvilínea. O valor do raio de curvatura define o tipo de trajetória. Para raios de curvatura entre 1,2 *m* a 30 *m*, o robô anda em trajetória curvilínea e acima de 30 *m* robô anda em trajetória retilínea. O raio mínimo de 1,2 *m* é imposto devido aos limites cinemáticos da junta 0 das pernas.

As margens angulares das juntas 0, 1, 2 e 3 são definidas pela diferença entre os valores máximos e mínimos dos ângulos das juntas.

Respeitar os limites cinemáticos do robô é a garantia que os movimentos programados sejam executados, otimizando a navegação do robô. A Tabela 5.1 mostra os limites operacionais do robô Guará.

*Tabela 5.1: Limites operacionais de navegação do robô Guará*

| Descrição | Valor |
|---|---|
| $h_1$ : altura máxima do obstáculo para transposição pela plataforma do robô | 200 mm |
| $l_1$ : comprimento entre as patas dianteiras e traseiras | 320 mm |
| $w_1$ : largura entre as patas dianteiras ou traseiras | 270 mm |
| $\theta_1$ : inclinação máxima do movimento lateral da plataforma do robô | 15º |
| $h_2$ : altura máxima do obstáculo para transposição com as patas | 50 mm |
| $l_2$ : comprimento máximo do obstáculo para transposição com as patas | 100 mm |
| $w_2$ : largura máxima do obstáculo para transposição pela plataforma do robô | 135 mm |
| $r_{c\min}$ : raio mínimo de curvatura para trajetória curvilínea | 1,20 m |
| $\phi_{\max}$ : ângulo máximo de guinada para trajetória curvilínea | 20º |
| $r_{c\max}$ : raio máximo de curvatura para definir a trajetória retilínea | 30,0 m |
| $g_x$ : margem de segurança do obstáculo, eixo X do referencial do robô | 50 mm |
| $g_y$ : margem de segurança do obstáculo, eixo Y do referencial do robô | 50 mm |
| $g_z$ : margem de segurança do obstáculo, eixo Z do referencial do robô | 20 mm |
| $d_{\min}$ : distância mínima usada para o controle de trajetória | 150 mm |
| $l_{pata}$ : comprimento da pata | 100 mm |
| $w_{pata}$ : largura da pata | 25 mm |
| $l_{passo}$ : comprimento médio de um passo completo do robô | 140 mm |
| $\phi_{\min}$ : ângulo mínimo usado para o controle de trajetória | 10º |

### 5.3.3 Condição de Obstáculos

O desvio ou transposição de obstáculos é uma das habilidades críticas para a aplicação de sistemas de robôs móveis e sua integração com o ambiente humano. A navegação

autônoma representa um alto nível de dificuldade, uma vez que requer a detecção e o desvio de obstáculos, simultaneamente, guiando o robô na direção do seu alvo [2].

Um obstáculo é representado por um conjunto de pequenos volumes agrupados, gerado através do processo de cubificação das imagens de profundidade.

O robô tem quatro situações a decidir face a um obstáculo: transpor o obstáculo; desviar-se do obstáculo à esquerda; desviar-se do obstáculo à direita ou dar um passo para trás. Esta decisão leva em conta as dimensões do objeto, sua posição relativa às quatro patas do robô e a direção preferencial em relação ao alvo destino do robô.

## 5.4 Árvore de Decisão

Para a implementação da Navegação Reativa foi escolhido um algoritmo baseado em Árvore de Decisão. As Árvores de Decisão representam um método de aprendizado simbólico amplamente utilizado para inferência indutiva. A vantagem deste método é a possibilidade de aproximar funções discretas robustas a dados incertos e permitir o uso de expressões disjuntas [53]. Elas representam uma maneira gráfica de visualizar as conseqüências de decisões atuais e futuras, bem como eventos aleatórios relacionados.

Este método mostra-se muito útil para a implementação da estratégia de controle para a navegação escolhida. As árvores de decisão classificam instâncias partindo do nó raiz, percorrendo os ramos, até atingir algum nó folha. Os nós representados por círculos indicam as tomadas de decisão enquanto que os nós representados por quadrados indicam as ações tomadas. Foi escolhida uma árvore de decisão binária, ou seja, para cada decisão existem apenas dois ramos. Os ramos representam os critérios.

As árvores de decisão binárias permitem implementar a estratégia *dividir-e-conquistar*, baseada em situações cujo objetivo é simplificar a dificuldade da navegação, porque existe uma divisão dos casos de navegação (situações) com ações de locomoção específicas para cada caso [54]. A Figura 5.2 mostra a implementação da árvore de decisão.

No caso do robô Guará, foi definido um conjunto completo de situações e ações que cercam todas as possíveis necessidades de navegação em ambiente com obstáculos.

### 5.4.1 Entradas da Árvore de Decisão

Representam os dados do sistema do robô, do sistema de visão e do alvo de destino do robô. Estes dados são as variáveis usadas pela árvore de decisão, e são enumerados a seguir:

a) **Limites de Estabilidade**: dados que garantam o equilíbrio do robô;
b) **Limites Cinemáticos**: dados que permitem uma navegação segura e estável;
c) **Odometria**: dados dos referenciais e de posição do robô e patas;
d) **Espaço de Movimento das Patas:** dados de trajetórias de vôo das patas;
e) **Descritores dos Obstáculos**: dados dos obstáculos da superfície de navegação do robô;
f) **Localização dos Obstáculos:** dados das posições dos obstáculos na superfície;
g) **Alvo de Destino do Robô:** ponto de término da trajetória do robô.

### 5.4.2 Critérios da Árvore de Decisão

Um critério representa uma situação em que o robô pode se encontrar durante a navegação. Cercar todas as situações de navegação garante ações corretas para que o robô atinja o seu alvo destino de forma segura, otimizada e eficiente. Após análise detalhada, os seguintes critérios baseados em decisão binária foram definidos:

1) **C1:** robô no alvo destino;
2) **C2:** robô em trajetória;
3) **C3:** robô em trajetória ótima;
4) **C4:** robô fora da trajetória ótima;
5) **C5:** robô fora da trajetória ótima com obstáculos na zona de segurança;
6) **C6:** robô fora da trajetória ótima sem obstáculos na zona de segurança;

7) **C7:** robô em trajetória ótima sem obstáculos na zona de segurança;
8) **C8:** robô em trajetória ótima com obstáculos na zona de segurança;
9) **C9:** robô com obstáculos a transpassar;
10) **C10:** robô com obstáculos a desviar-se;
11) **C11:** robô com obstáculos fora do espaço de movimento das patas;
12) **C12:** robô com obstáculos no espaço de movimento das patas;
13) **C13:** robô em transposição dos obstáculos;
14) **C14:** robô em equalização para transposição dos obstáculos;
15) **C15:** robô com obstáculos com áreas livres;
16) **C16:** robô com obstáculos e com área livre para desviar-se à esquerda;
17) **C17:** robô com obstáculos e com área livre para desviar-se à direita;
18) **C18:** robô com obstáculos sem áreas livres;
19) **C19:** robô fora da trajetória ótima na direção correta ao alvo;
20) **C20:** robô fora da trajetória ótima na direção errada ao alvo;
21) **C21:** robô fora da trajetória ótima na direção correta ao alvo para virar à esquerda;
22) **C22:** robô fora da trajetória ótima na direção correta ao alvo para virar à direita;
23) **C23:** robô fora da trajetória ótima na direção errada ao alvo para virar à esquerda;
24) **C24:** robô fora da trajetória ótima na direção errada ao alvo para virar à direita.

### 5.4.3 Ações da Árvore de Decisão

Em cada situação, uma ação individual resolve uma tarefa da navegação reativa, cujo objetivo é evitar os obstáculos ou transpassá-los, enquanto o robô se locomove em direção ao alvo de destino de forma otimizada, reduzindo ao máximo a distância de sua trajetória.

1) **A1:** robô pára;
2) **A2:** robô dá um passo à frente em trajetória retilínea;
3) **A3:** robô faz a equalização das 4 patas;
4) **A4:** robô dá um passo à frente em forma de arco à esquerda;
5) **A5:** robô dá um passo à frente em forma de arco à direita;
6) **A6:** robô recua um passo em trajetória retilínea.

A Figura 5.2 mostra graficamente o método de navegação usado pelo robô Guará.

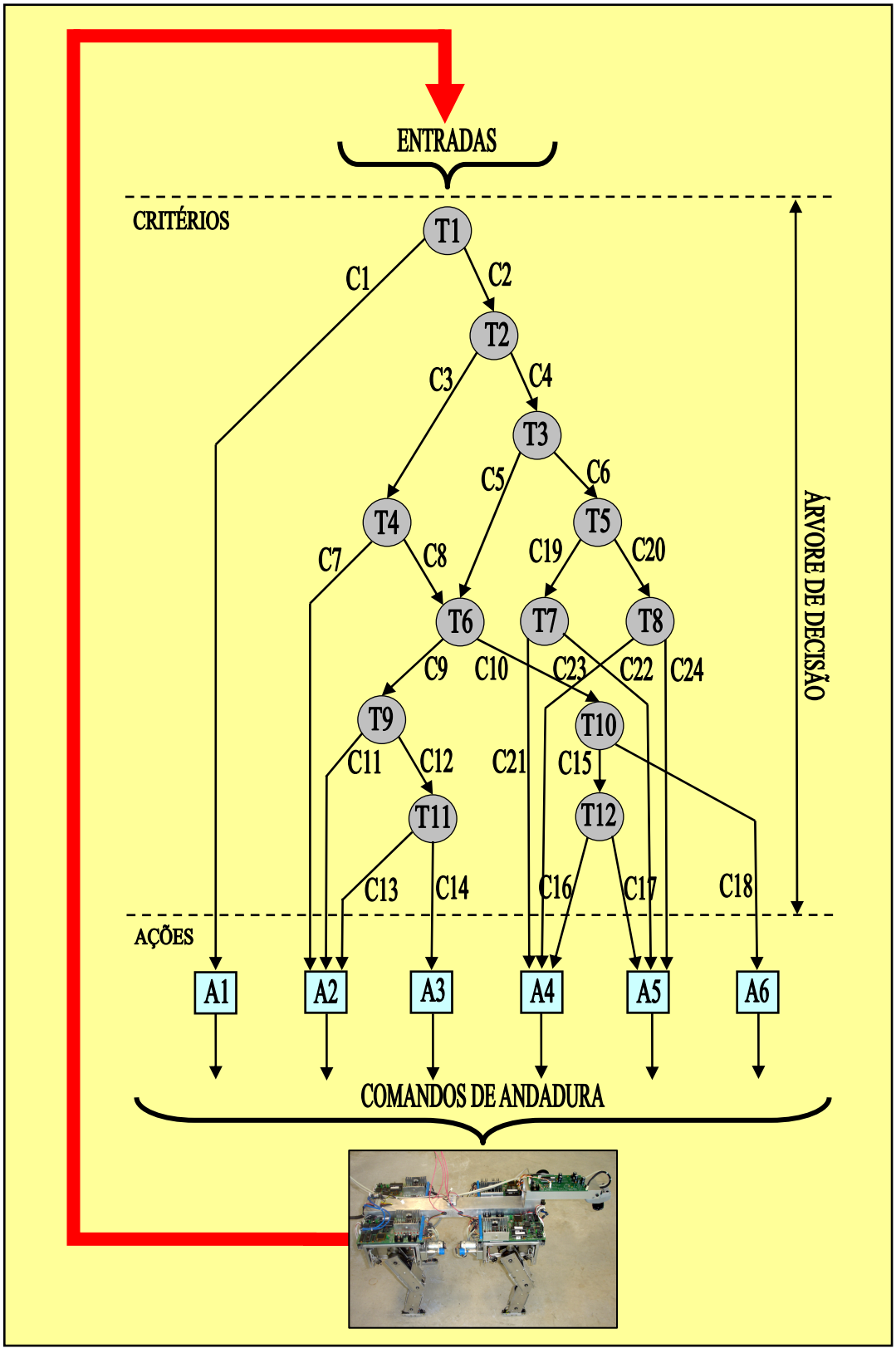


*Figura 5.2: Método da navegação reativa implementado para o robô Guará.*

O método implementado de navegação reativa é baseado em dados geométricos de posicionamento do robô e dos obstáculos. Para viabilizar os seis comandos definidos de andadura, o robô é provido com um conjunto de dados que o norteará na execução do movimento.

A Figura 5.3 mostra as possíveis situações e conseqüentes ações do robô Guará, face aos obstáculos em sua trajetória, em direção ao alvo de destino.

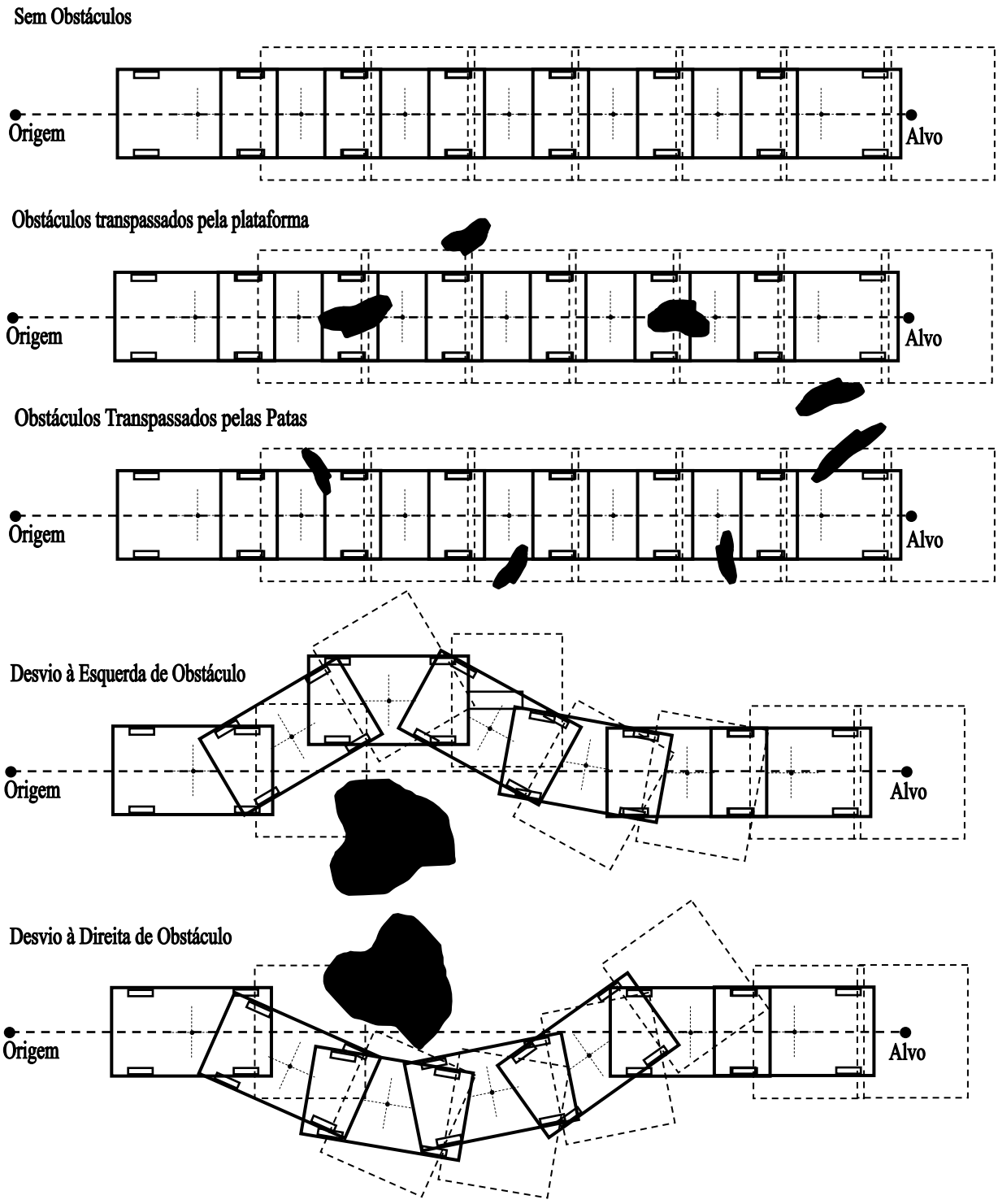


*Figura 5.3: Exemplos de situações e ações do robô Guará.*

## 5.5 Tomadas de Decisão Implementadas

São inúmeras as possibilidades de posicionamento de obstáculos em relação às 4 patas do robô. O algoritmo de tomada de decisão tende a ser mais complexo à medida que a quantidade de obstáculos mapeados aumenta.

O algoritmo implementado tem um conjunto reduzido de tomadas de decisão. O robô parte de um referencial fixo, e é programado para atingir um alvo de destino. No caminho ele confronta com os obstáculos e faz o mapeamento. As estratégias de tomadas de decisão são descritas a seguir.

### 5.5.1 Desvio à Esquerda de Obstáculo Intransponível

A Figura 5.4 ilustra a situação de o robô encontrar um obstáculo à direita. Seja o obstáculo $O_i$ com altura $h_{Oi}$, largura $w_{Oi}$, localizado dentro dos círculos de procura $C_{0i}$ e $C_{2i}$. Este obstáculo é instransponível quando uma das 2 situações forem verdadeiras:

- ✓ $h_{Oi} > h_1 - g_z$,
- ✓ $w_{Oi} > w_2 - g_y$,

onde $h_1$, $g_z$, $w_2$ e $g_y$ são limites operacionais descritos na Tabela 5.1.

Cada quadro de imagem é percorrido por aproximadamente 3 passos completos do robô. Caso o obstáculo seja detectado no primeiro passo e ainda não tenha sido mapeado totalmente, os seus descritores estão parcialmente definidos. A tomada de decisão pode interpretar que este obstáculo é transponível e o robô continua sua trajetória. Nos passos seguintes, o obstáculo é mapeado totalmente, e este obstáculo é rotulado como intransponível.

No caso do obstáculo intransponível, para que o desvio seja possível, é verificada a distância mais próxima do obstáculo até a pata 2, $d_{2p}$. Para se desviar, esta distância $d_{2p}$

deve ser maior que o raio do círculo de procura $C_{2i}$. Caso contrário, o robô dá um passo para trás, aumentando a distância até o obstáculo e o processo se repete.

Seja o instante *i* de análise. O vetor ${}^{o}\mathbf{p}_r = \begin{bmatrix} {}^{o}x_r & {}^{o}y_r & {}^{o}z_r \end{bmatrix}^T$ indica a posição do referencial do robô, ${}^{o}\mathbf{p}_{03} = \begin{bmatrix} {}^{o}x_{03} & {}^{o}y_{03} & {}^{o}z_{03} \end{bmatrix}^T$ a posição pata 0, ${}^{o}\mathbf{p}_{13} = \begin{bmatrix} {}^{o}x_{13} & {}^{o}y_{13} & {}^{o}z_{13} \end{bmatrix}^T$ a posição da pata 1, ${}^{o}\mathbf{p}_{23} = \begin{bmatrix} {}^{o}x_{23} & {}^{o}y_{23} & {}^{o}z_{23} \end{bmatrix}^T$ a posição da pata 2 e ${}^{o}\mathbf{p}_{33} = \begin{bmatrix} {}^{o}x_{33} & {}^{o}y_{33} & {}^{o}z_{33} \end{bmatrix}^T$ a posição da pata 3, todos em relação ao referencial da origem. O vetor ${}^{o}\mathbf{p}_e = \begin{bmatrix} {}^{o}x_e & {}^{o}y_e & {}^{o}z_e \end{bmatrix}^T$ representa o ponto mais próximo à esquerda do obstáculo, mantida uma guarda de segurança ao seu redor, que é definida pelos valores de $g_x$, $g_y$ e $g_z$. O vetor ${}^{o}\mathbf{p}_{e2} = \begin{bmatrix} {}^{o}x_{e2} & {}^{o}y_{e2} & {}^{o}z_{e2} \end{bmatrix}^T$ representa o ponto médio entre os vetores ${}^{o}\mathbf{p}_e$ e ${}^{o}\mathbf{p}_{23}$, e o vetor ${}^{o}\mathbf{p}_m = \begin{bmatrix} {}^{o}x_m & {}^{o}y_m & {}^{o}z_m \end{bmatrix}^T$ representa o ponto médio entre os vetores ${}^{o}\mathbf{p}_{03}$ e ${}^{o}\mathbf{p}_{13}$.

Sejam as retas *r*, *s* e *t* definidas no plano $z = 0$ do sistema de coordenadas do mundo. A reta *r* contém os pontos ${}^{o}\mathbf{p}_{e2}$ e ${}^{o}\mathbf{p}_c$, a reta *s* contém os pontos ${}^{o}\mathbf{p}_r$, ${}^{o}\mathbf{p}_m$ e ${}^{o}\mathbf{p}_c$ e a reta *t* contém os pontos ${}^{o}\mathbf{p}_e$, ${}^{o}\mathbf{p}_{23}$ e ${}^{o}\mathbf{p}_{e2}$. As retas *r* e *t* são perpendiculares.

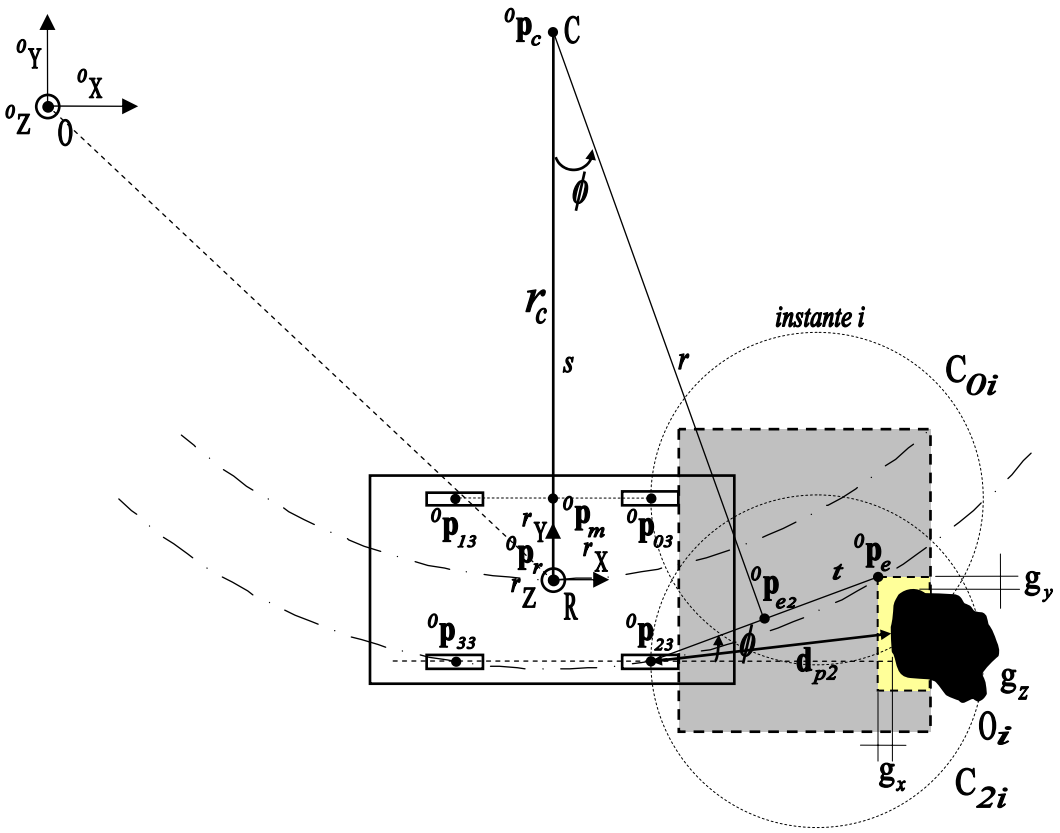


*Figura 5.4: Desvio à esquerda de um obstáculo intransponível.*

O objetivo é determinar o raio $r_c$ de curvatura do robô e o ângulo $\phi$ de guinada para que o robô possa se desviar do obstáculo de forma tangencial, sem que as patas o toquem, respeitando a margem de segurança.

Sejam $m_t$ e $m_r$ coeficientes angulares das retas *t* e *r* respectivamente, sendo que $t \perp r$. Logo,

$$\begin{aligned} m_t &= \frac{^o x_e - {}^o x_{23}}{^o y_e - {}^o y_{23}}, \forall\, ^o y_e \neq {}^o y_{23}, \\ m_t . m_r &= -1, \\ m_r &= \frac{^o y_{23} - {}^o y_e}{^o x_{23} - {}^o x_e}, \forall\, ^o x_{23} \neq {}^o x_e. \end{aligned} \tag{5.1}$$

Os casos de singularidade, quando o ângulo de declividade das retas é $\pm 90^o$, são tratados como casos particulares. A equação da reta *r* é dada por

$$x - {}^o x_{e2} = m_r . (y - {}^o y_{e2}), \tag{5.2}$$

onde $(x, y)$ representa um ponto qualquer pertencente à reta *r*.

Seja $m_s$ o coeficiente angular da reta *s*. Logo,

$$m_s = \frac{{}^o x_m - {}^o x_r}{{}^o y_m - {}^o y_r}, \forall {}^o y_m \neq {}^o y_r, \tag{5.3}$$

e a equação da reta *s é dada por*

$$x - {}^o x_r = m_s.(y - {}^o y_r), \tag{5.4}$$

onde $(x, y)$ representa um ponto qualquer pertencente à reta *s*.

As retas *r* e *s* se interceptam no ponto *C*. Para descobrir este ponto, basta resolver o sistema de equações

$$\begin{cases} x - {}^o x_{e2} = m_r.(y - {}^o y_{e2}) \\ x - {}^o x_r = m_s.(y - {}^o y_r), \end{cases} \tag{5.5}$$

encontrando assim o vetor ${}^o\mathbf{p}_c = \begin{bmatrix} {}^o x_c & {}^o y_c & {}^o z_c \end{bmatrix}^T$.

O raio $r_c$ de curvatura do robô é dado pela distância euclidiana entre os vetores ${}^o\mathbf{p}_c$ e ${}^o\mathbf{p}_r$ no plano $z = 0$, logo

$$r_c = \sqrt{({}^0 x_c - {}^0 x_r)^2 + ({}^0 y_c - {}^0 y_r)^2}. \tag{5.6}$$

O ângulo $\phi$ de guinada é dado por

$$\phi = \arctan\left( \frac{{}^0 y_e - {}^0 y_{23}}{{}^0 x_e - {}^0 x_{23}} \right). \tag{5.7}$$

### 5.5.2 Desvio à Direita de Obstáculo Intransponível

A Figura 5.5 mostra a situação em que o robô se depara com um obstáculo intransponível à esquerda. Esta situação é semelhante àquela descrita no item 5.5.1. As condições analisadas para tomada de decisão são as mesmas, sendo que a distância mais próxima do obstáculo até a pata 0 é dada por $d_{p0}$.

Seja o instante *i* de análise; o vetor ${}^o\mathbf{p}_r$ indica a posição do referencial do robô, ${}^o\mathbf{p}_{03}$ a posição da pata 0, ${}^o\mathbf{p}_{13}$ a posição da pata 1, ${}^o\mathbf{p}_{23}$ a posição da pata 2 e ${}^o\mathbf{p}_{33}$ a posição da pata 3, todos em relação ao referencial do mundo e projetados no plano $z = 0$. O vetor

${}^{o}\mathbf{p}_{d} = \begin{bmatrix} {}^{o}x_{d} & {}^{o}y_{d} & {}^{o}z_{d} \end{bmatrix}^{T}$ representa o ponto mais próximo à direita do obstáculo, mantida uma guarda de segurança ao seu redor, que é definida pelos valores de $g_x$, $g_y$ e $g_z$. O vetor ${}^{o}\mathbf{p}_{d0} = \begin{bmatrix} {}^{o}x_{d0} & {}^{o}y_{d0} & {}^{o}z_{d0} \end{bmatrix}^{T}$ representa o ponto médio entre os vetores ${}^{o}\mathbf{p}_{d}$ e ${}^{o}\mathbf{p}_{03}$, e o vetor ${}^{o}\mathbf{p}_{m} = \begin{bmatrix} {}^{o}x_{m} & {}^{o}y_{m} & {}^{o}z_{m} \end{bmatrix}^{T}$ representa o ponto médio entre os vetores ${}^{o}\mathbf{p}_{23}$ e ${}^{o}\mathbf{p}_{33}$.

Sejam as retas *r*, *s* e *t* definidas no plano $z = 0$ do sistema de coordenadas da origem. A reta *r* contém os pontos ${}^{o}\mathbf{p}_{d0}$ e ${}^{o}\mathbf{p}_{c}$, a reta *s* contém os pontos ${}^{o}\mathbf{p}_{r}$, ${}^{o}\mathbf{p}_{m}$ e ${}^{o}\mathbf{p}_{c}$ e a reta *t* contém os pontos ${}^{o}\mathbf{p}_{d}$, ${}^{o}\mathbf{p}_{03}$ e ${}^{o}\mathbf{p}_{d0}$. As retas *r* e *t* são perpendiculares.

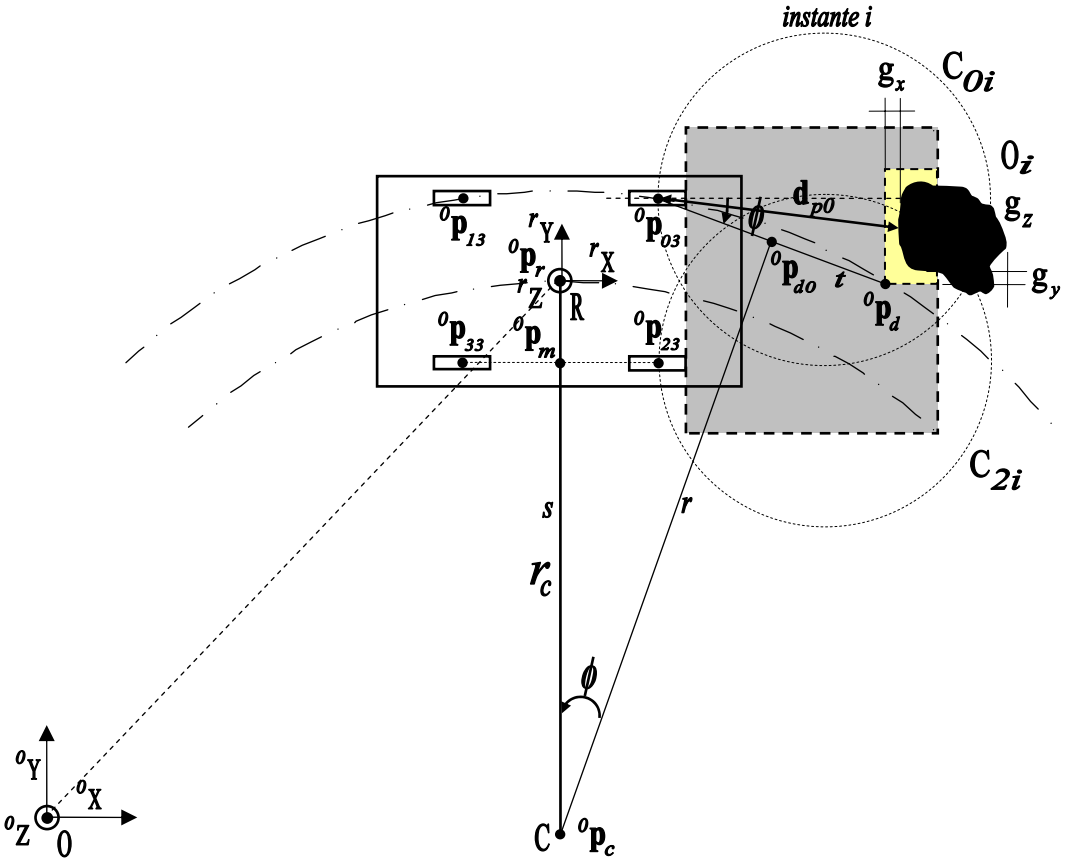


*Figura 5.5: Desvio à direita de um obstáculo intransponível.*

O objetivo é determinar geometricamente o raio $r_c$ de curvatura do robô e o ângulo $\phi$ de guinada para que o robô possa se desviar do obstáculo, sem que as patas o toquem, respeitando a margem de segurança.

Sejam $m_t$ e $m_r$ os coeficientes angulares das retas *t* e *r* respectivamente, sendo que $t \perp r$. Logo,

$$\begin{aligned} m_t &= \frac{{}^o x_d - {}^o x_{03}}{{}^o y_d - {}^o y_{03}}, \forall\, {}^o y_d \neq {}^o y_{03} \\ m_t . m_r &= -1, \\ m_r &= \frac{{}^o y_{03} - {}^o y_d}{{}^o x_{03} - {}^o x_d}, \forall\, {}^o x_{03} \neq {}^o x_d . \end{aligned} \tag{5.8}$$

Os casos de singularidade, quando o ângulo de declividade das retas é $\pm 90^o$, são tratados como casos particulares. A equação da reta *r é dada por*

$$x - {}^o x_{d0} = m_r . (y - {}^o y_{d0}), \tag{5.9}$$

onde $(x, y)$ representa um ponto qualquer pertencente à reta *r*.

Seja $m_s$ o coeficiente angular da reta *s*. Logo,

$$m_s = \frac{{}^o x_m - {}^o x_r}{{}^o y_m - {}^o y_r}, \forall\, {}^o y_m \neq {}^o y_r, \tag{5.10}$$

e a equação da reta *s* é dada por

$$x - {}^o x_r = m_s . (y - {}^o y_r), \tag{5.11}$$

onde $(x, y)$ representa um ponto qualquer pertencente à reta *s*.

As retas *r* e *s* se interceptam no ponto *C* e para descobrir este ponto, basta resolver o sistema de equações

$$\begin{cases} x - {}^o x_{d0} = m_r . (y - {}^o y_{d0}) \\ x - {}^o x_r = m_s . (y - {}^o y_r), \end{cases} \tag{5.12}$$

encontrando assim o vetor ${}^o\mathbf{p}_c = \begin{bmatrix} {}^o x_c & {}^o y_c & {}^o z_c \end{bmatrix}^T$.

O raio $r_c$ de curvatura do robô é dado pela distância euclidiana entre os vetores ${}^o\mathbf{p}_c$ e ${}^o\mathbf{p}_r$ no plano $z = 0$, logo

$$r_c = \sqrt{({}^0 x_c - {}^0 x_r)^2 + ({}^0 y_c - {}^0 y_r)^2}. \tag{5.13}$$

O ângulo $\phi$ de guinada é dado por

$$\phi = \arctan\left( \frac{{}^0 y_d - {}^0 y_{03}}{{}^0 x_d - {}^0 x_{03}} \right). \tag{5.14}$$

### 5.5.3 Transposição Direta de Obstáculo sob o Robô

Seja o obstáculo $O_i$ com altura $h_{Oi}$, largura $w_{Oi}$, localizado dentro dos círculos de procura $C_{0i}$, $C_{2i}$ e $C_{13i}$. É verificada a condição de que este obstáculo não está na trajetória das patas, analisando as distâncias frontais $d_{0f}$ e $d_{2f}$, cujos valores devem ser nulos. Seja $d_e$, a distância normal do ponto ${}^{o}\mathbf{p}_{SE}$ do obstáculo até a reta que passa pelas patas 0 e 1. Seja $d_d$, a distância normal do ponto ${}^{o}\mathbf{p}_{SD}$ do obstáculo até a reta que passa pelas patas 2 e 3. As distâncias $d_{0d}$ e $d_{1d}$ são as distâncias laterais direita da pata 0 e 1 até o obstáculo, respectivamente, e a distância $d_{2e}$ e $d_{3e}$ são as distâncias laterais esquerda das patas 2 e 3 até o obstáculo, respectivamente. Este obstáculo é transponível, com o robô em trajetória retilínea, quando as seguintes condições são satisfeitas:

- ✓ $h_{Oi} <= h_1 - g_z$,
- ✓ $w_{Oi} <= w_2 - g_y$,
- ✓ $d_e > g_y$,
- ✓ $d_d > g_y$,
- ✓ $d_{0d} > g_y$,
- ✓ $d_{1d} > g_y$,
- ✓ $d_{2e} > g_y$,
- ✓ $d_{3e} > g_y$,

onde $h_1$, $g_z$, $w_2$ e $g_y$ são limites operacionais descritos na Tabela 5.1. A Figura 5.6 mostra esta situação.

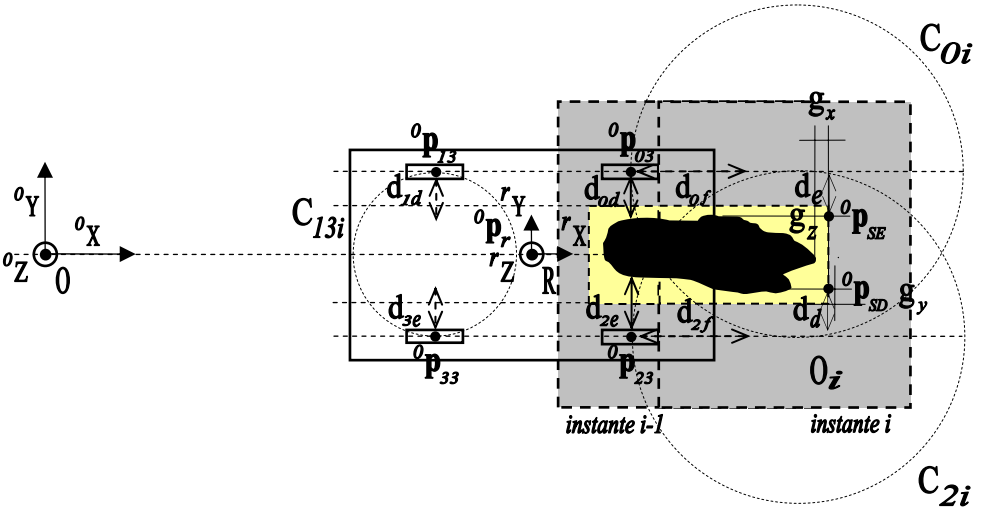


*Figura 5.6: Transposição direta de obstáculo sob o robô.*

### 5.5.4 Transposição de Obstáculo pela Pata em Vôo

Seja o obstáculo $O_i$ com altura $h_{Oi}$, largura $w_{Oi}$ e comprimento $l_{Oi}$, localizado dentro do círculo de procura $C_{0i}$ com raio $r_{Coi}$. Seja $d_{0f}$ a distância frontal da pata 0 até o obstáculo. Para que um obstáculo possa ser transposto pelas patas do robô Guará, este obstáculo necessita estar na direção da trajetória das patas, na frente das patas frontais ou traseiras. A Figura 5.7 mostra esta situação, para o caso da pata 0. Este obstáculo é transposto pelas patas quando as seguintes situações são satisfeitas:

- ✓ $h_{Oi} <= h_2 - g_z$,
- ✓ $l_{Oi} <= l_2 - g_x$,
- ✓ $w_{Oi} < r_{Coi}$,
- ✓ $d_{0f} > g_y$,

onde $h_2$, $g_z$, $l_2$ e $g_x$ são limites operacionais descritos na Tabela 5.1.

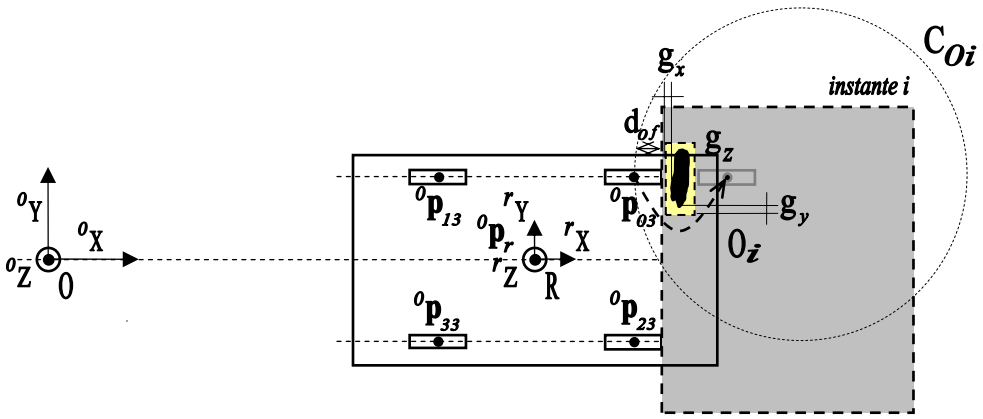


*Figura 5.7: Transposição de obstáculo pela pata em vôo.*

Os limites operacionais de elevação e alcance das patas em vôo são estreitos, transpondo apenas obstáculos baixos e curtos, conforme descritos na Tabela 5.1. Durante a transposição pelas patas, caso o objeto esteja afastado, o robô faz uma equalização das patas para ficar o mais próximo possível da pata a ser transposta.

### 5.5.5 Correção da Trajetória rumo ao Alvo de Destino

O robô muda a sua trajetória programada ao se desviar de algum obstáculo. Este desvio é tanto maior quanto maior for o tamanho do obstáculo ao seu redor. O sistema de controle corrige a trajetória do robô, trazendo-o para a trajetória ótima. Existem quatro possibilidades de posicionamento do robô quando fora da trajetória ótima. A Figura 5.8 mostra as quatro possibilidades, como segue:

a) robô posicionado na direção correta, à esquerda do alvo;
b) robô posicionado na direção correta, à direita do alvo;
c) robô posicionado na direção errada, à esquerda do alvo;
d) robô posicionado na direção errada, à direita do alvo.

Sejam $r_{ao}$, a reta que passa pelo referencial da origem *O* e pelo ponto do alvo de destino ${}^{o}\mathbf{p}_a = \begin{bmatrix} {}^{o}x_a & {}^{o}y_a & {}^{o}z_a \end{bmatrix}^T$, e $r_{ar}$, a reta que passa e pelo referencial do robô *R* e pelo ponto

do alvo de destino ${}^{o}\mathbf{p}_{a}$. Define-se o ângulo $\phi_{ra} = \arctan(m_{ar}) - \arctan(m_{ao})$, onde $m_{ar}$ é o coeficiente angular da reta $r_{ar}$ e $m_{ao}$ é o coeficiente angular da reta $r_{ao}$.

Para se descobrir se o robô se encontra à esquerda ou à direita do alvo, é analisado o valor de $\phi_{ra}$. Se $0^o \leq \phi_{ra} < 180^o$, o robô se encontra à esquerda do alvo e se $0^o > \phi_{ra} \geq -180^o$, o robô se encontra à direita do alvo.

Seja $r_r$ a reta que passa pelo referencial *R* e paralela ao eixo ${}^{r}X$. Define-se o ângulo $\phi_r = \arctan(m_r) - \arctan(m_{ao})$, onde $m_r$ é o coeficiente angular da reta $r_r$ e $m_{ao}$ é o coeficiente angular da reta $r_{ao}$.

Para se descobrir se o robô se encontra na direção certa ou errada do alvo é analisado o valor e $\phi_r$. Se $-90^o < \phi_r < 90^o$, o robô se encontra na direção certa do alvo, e se $\phi_r \geq 90^o$ ou $\phi_r \leq -90^o$, o robô se encontra na direção contrária ao alvo.

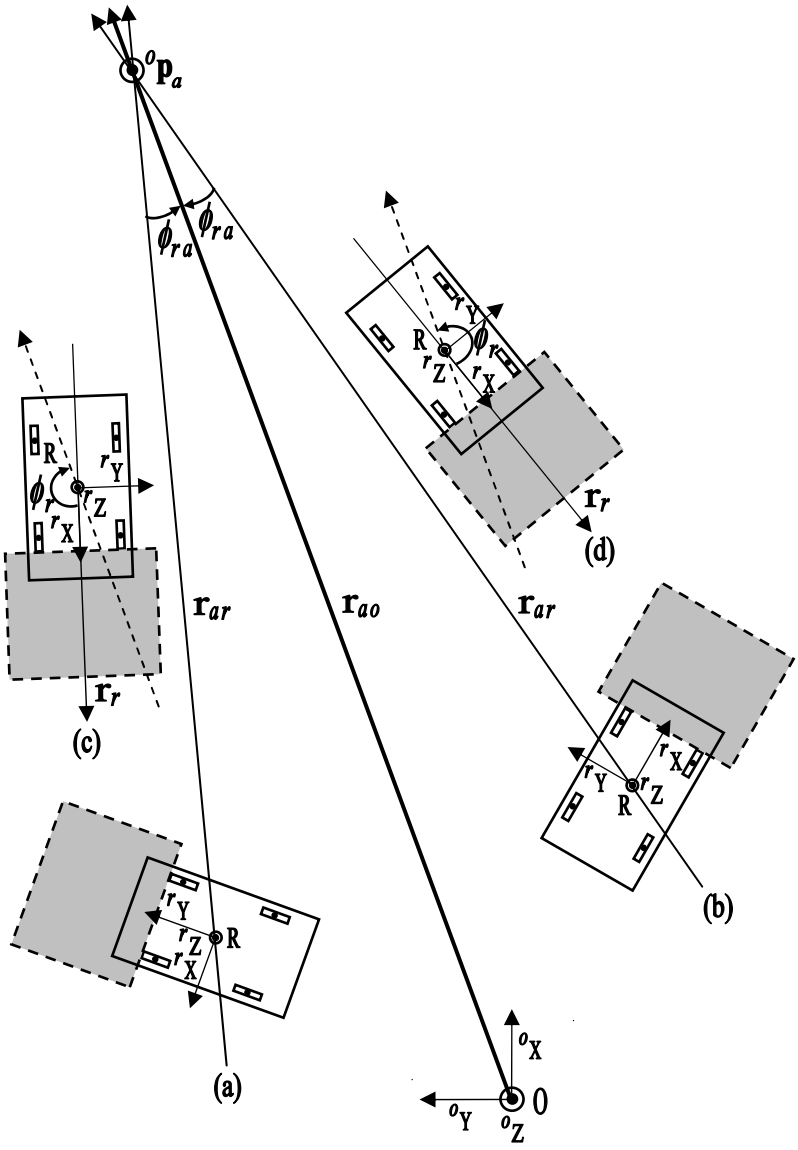


*Figura 5.8: Situações de posicionamento do robô perante o alvo de destino.*

A Figura 5.9 mostra uma situação em que o robô, ao sair da origem, se deparou com obstáculos intransponíveis à esquerda, desviando-se para a direita. No instante *i,* o sistema de controle constata que não há mais obstáculos no entorno do robô, portanto um procedimento para trazer o robô para a trajetória ótima é iniciado.

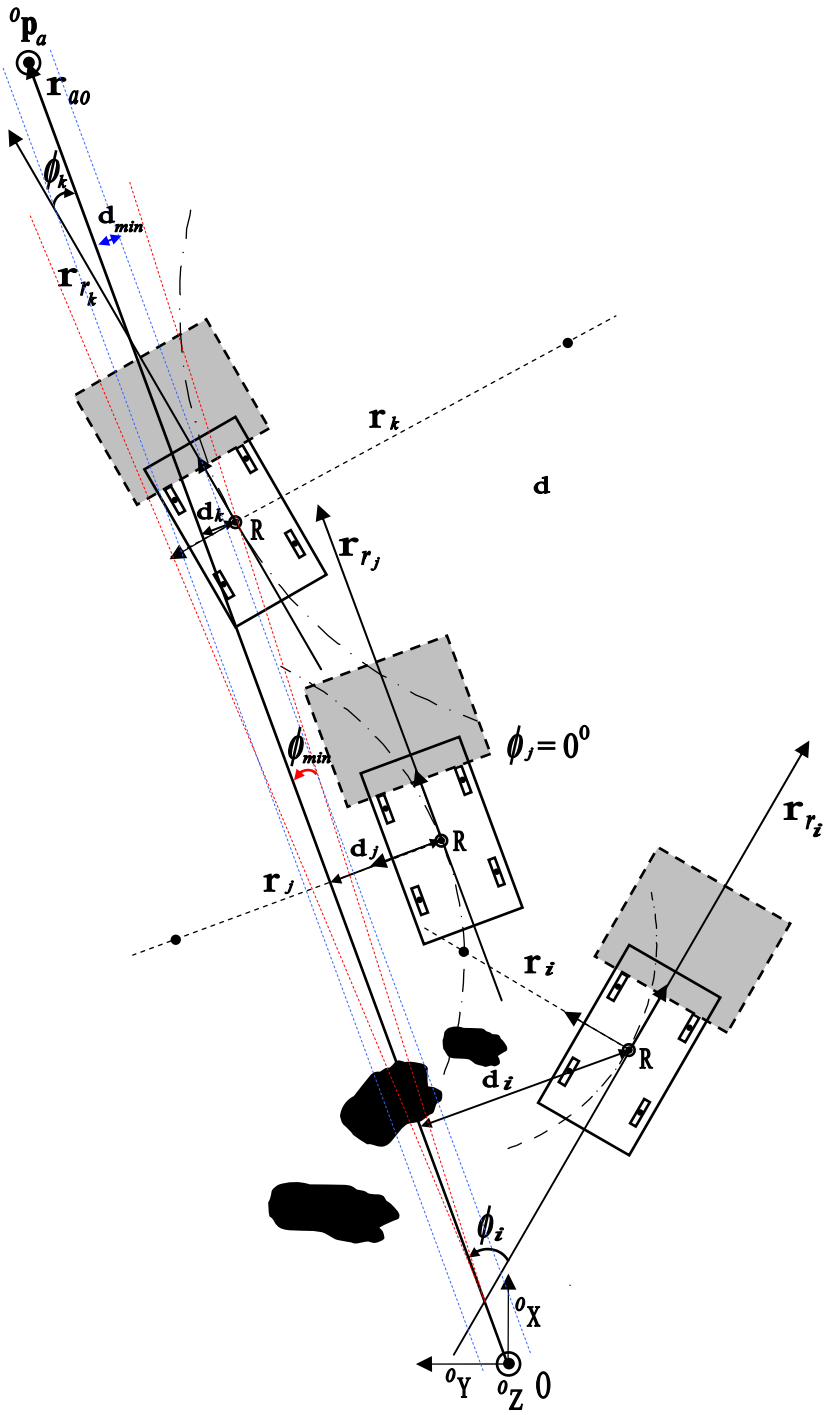


*Figura 5.9: Correção da trajetória na ausência de obstáculos com robô à direita do alvo.*

Seja o raio de curvatura $r_i$. Este raio varia de $r_{c\min}$ a $r_{c\max}$, conforme mostrado na Tabela 5.1. Seja $d_i$ a distância perpendicular do referencial *R* até a reta da trajetória $r_{ao}$ e $\phi_i$ o

ângulo de inclinação entre as retas $r_{ao}$ e $r_{ri}$. O raio de curvatura $r_i$ é tanto menor quanto maior for a distância do robô em relação à trajetória e quanto maior for a inclinação do robô em relação à trajetória. À medida que o robô se aproxima da trajetória, nos instantes *j* e *k*, respectivamente, a distância e a inclinação diminuem, e o raio de curvatura aumenta, tendendo o robô para uma locomoção retilínea. No instante *j*, o robô se encontra paralelo à trajetória, e, portanto o ângulo de inclinação vale zero. No instante *k*, a inclinação muda de sentido, e, conseqüentemente o raio de curvatura também muda de sentido.

Por convenção, se o robô gira à esquerda o raio de curvatura é positivo, enquanto que no giro à direita o raio de curvatura é negativo. O ângulo $\phi_i$ é definido por $\phi_i = \arctan(m_{ri}) - \arctan(m_{ao})$, onde $m_{ri}$ é o coeficiente angular da reta $r_{ri}$ e $m_{ao}$ é o coeficiente angular da reta $r_{ao}$.

Dois parâmetros de histerese são definidos: $d_{\min}$ é a distância mínima que indica que o robô está próximo da trajetória ótima, e $\phi_{\min}$ é ângulo mínimo de inclinação, tanto positivo quanto negativo, entre as retas, e indica que a direção do robô está próxima da direção da trajetória ótima.

Para o caso mostrado na Figura 5.9, a seguinte função para o cálculo do raio de curvatura $r_i$ foi definida:

$$r_i = \begin{cases} r_{c\min}, \quad se \quad \forall d_i, 90^\circ > \phi_i > \phi_{\min} \; (a) \\ -r_{c\min}, \quad se \quad d_i \le d_{\min}, -90^\circ < \phi_i < -\phi_{\min} \; (b) \\ \dfrac{-r_{c\max}}{2}, \quad se \quad d_i > d_{\min}, -90^\circ < \phi_i < -\phi_{\min} \; (c) \\ \dfrac{(r_{c\max} - r_{c\min})}{\phi_{\min}^2}.(\phi_i - \phi_{\min})^2 + r_{c\min}, \quad se \quad d_i \le d_{\min}, 0^\circ \le \phi_i \le \phi_{\min} \; (d) \\ \dfrac{(-r_{c\max} + r_{c\min})}{\phi_{\min}^2}.(\phi_i + \phi_{\min})^2 - r_{c\min}, \quad se \quad d_i \le d_{\min}, 0^\circ > \phi_i \ge -\phi_{\min} \; (e) \\ r_{c\min}, \quad se \quad d_i > d_{\min}, \phi_{\min} > \phi_i > -\phi_{\min} \; (f) \end{cases} \tag{5.15}$$

A função (5.15) foi definida de modo a calcular o raio de curvatura quando o robô se encontra à direita do alvo, da seguinte maneira:

a) caso o ângulo de inclinação $\phi_i$ esteja acima do valor mínimo $\phi_{\min}$, mas abaixo de $90^\circ$, independente da distância $d_i$ normal à reta $r_{ao}$, o raio deve ser o mínimo $r_{c\min}$ para dar a maior guinada à esquerda e trazer o robô para a trajetória ótima;

b) caso o robô esteja próximo da trajetória ótima ($d_i \leq d_{\min}$) e o ângulo de inclinação esteja abaixo de $-\phi_{\min}$, indicando que o robô está com a sua orientação muito inclinada à esquerda, é necessário dar uma guinada à direita, com $-r_{c\min}$;

c) caso o robô esteja mais afastado da trajetória ($d_i > d_{\min}$), mas o ângulo de inclinação esteja abaixo de $-\phi_{\min}$, indicando que o robô está com a sua orientação muito inclinada à esquerda, é necessário dar uma guinada à direita com a metade do raio máximo ($-r_{c\max}/2$);

d) caso o robô esteja próximo da trajetória ótima ($d_i \leq d_{\min}$) e o ângulo de inclinação $\phi_i$ esteja entre os valores $0$ e $\phi_{\min}$, indicando uma inclinação pequena, o raio é calculado segundo a equação (5.15), item *d* e a Figura 5.10 mostra graficamente este comportamento (curva do primeiro quadrante). O raio é positivo, indicando uma rotação da plataforma do robô para a esquerda;

e) caso o robô esteja próximo da trajetória ótima ($d_i \leq d_{\min}$) e o ângulo de inclinação $\phi_i$ esteja entre os valores $-\phi_{\min}$ e $0$, indicando uma inclinação pequena, o raio é calculado segundo a equação (5.15), item *e* e a Figura 5.10 mostra graficamente este comportamento (curva do terceiro quadrante). O raio é negativo, indicando uma rotação da plataforma do robô para a direita;

f) caso o robô esteja distante e o ângulo de declividade $\phi_i$ esteja entre os valores $-\phi_{\min}$ e $\phi_{\min}$, o raio deve ser o mínimo, $r_{c\min}$, para dar a maior guinada à esquerda e trazer o robô para a trajetória ótima.

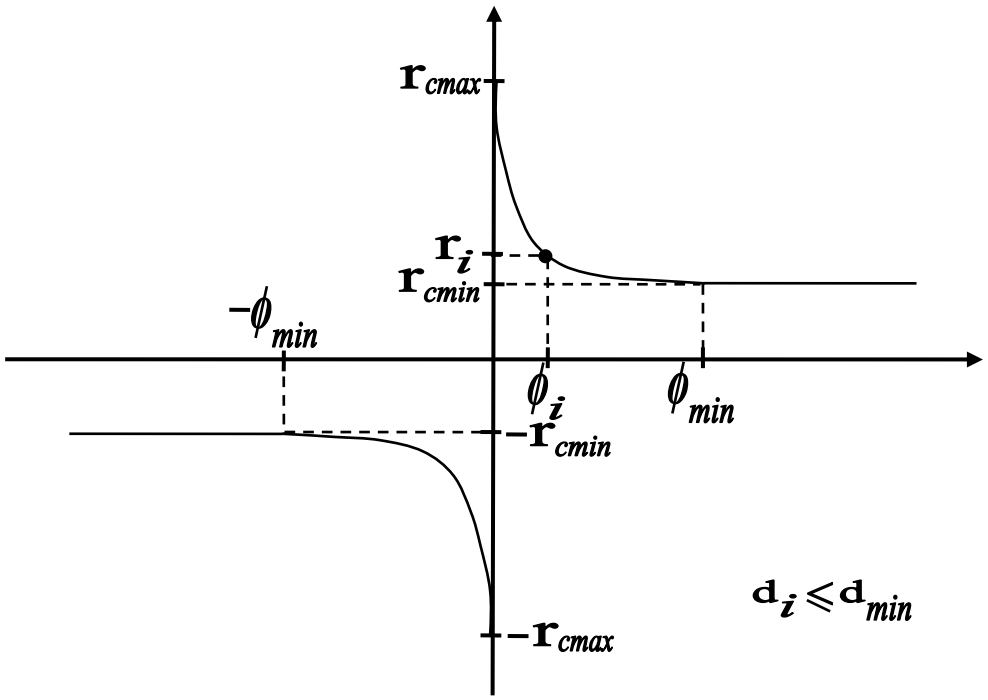


*Figura 5.10: Gráfico da função de controle para geração do raio de curvatura.*

A Figura 5.11 mostra uma situação em que o robô, ao sair da origem, deparou-se com obstáculos intransponíveis à direita, desviando-se para a esquerda. No instante *i,* o sistema de controle constata que não há mais obstáculos no entorno do robô, portanto um procedimento para trazer o robô para a trajetória ótima é iniciado.

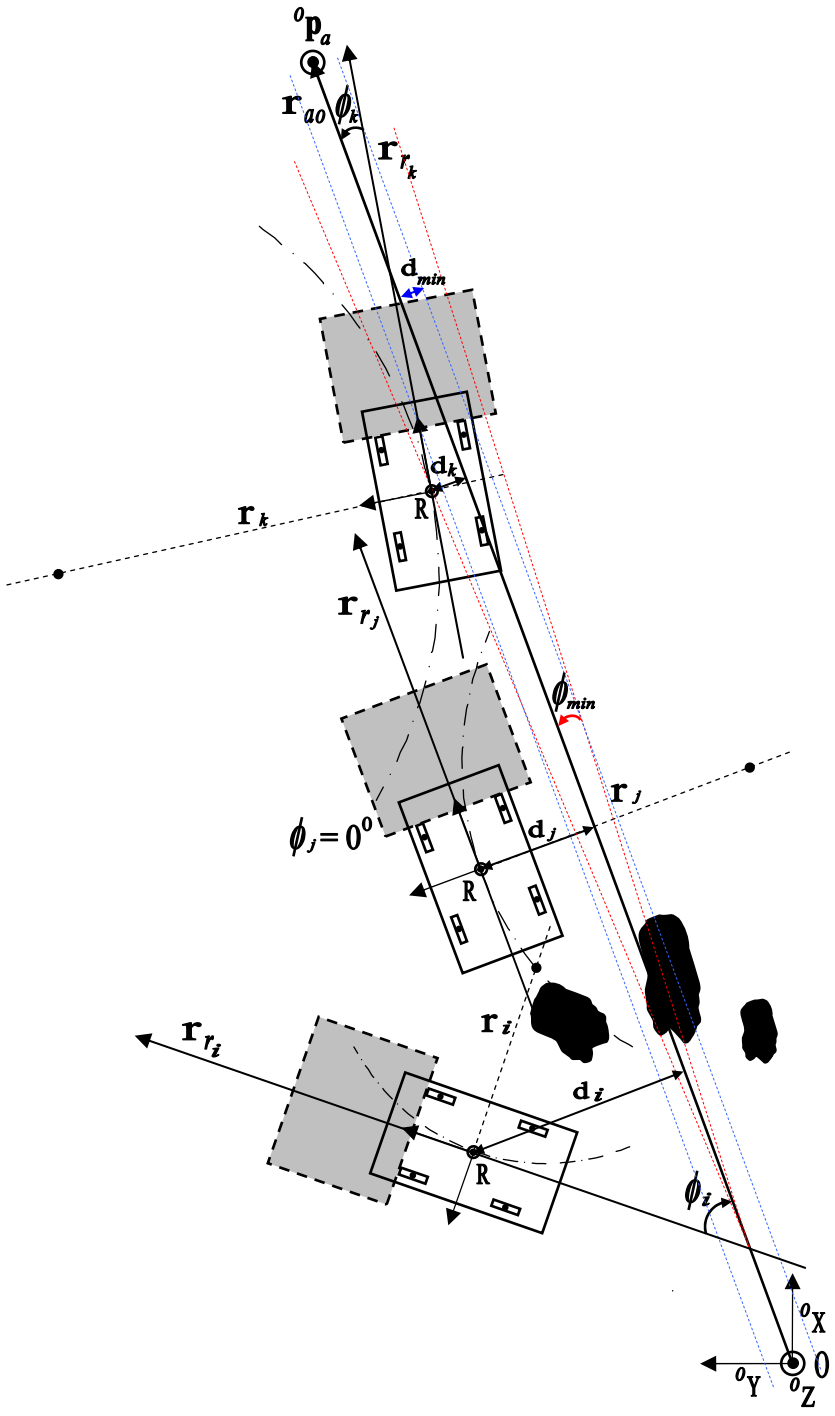


*Figura 5.11: Correção da trajetória na ausência de obstáculo com robô à esquerda do alvo.*

Para o caso mostrado na Figura 5.11 , a seguinte função para o cálculo do raio de curvatura $r_i$ foi definida:

$$r_i = \begin{cases} -r_{c\min}, \quad se \quad \forall d_i, -90^\circ < \phi_i < -\phi_{\min} \; (a) \\ r_{c\min}, \quad se \quad d_i \le d_{\min}, 90^\circ > \phi_i > \phi_{\min} \; (b) \\ \dfrac{r_{c\max}}{2}, \quad se \quad d_i > d_{\min}, 90^\circ > \phi_i > \phi_{\min} \; (c) \\ \dfrac{(r_{c\max} - r_{c\min})}{\phi_{\min}^2}.(\phi_i - \phi_{\min})^2 + r_{c\min}, \quad se \quad d_i \le d_{\min}, 0^\circ \le \phi_i \le \phi_{\min} \; (d) \\ \dfrac{(-r_{c\max} + r_{c\min})}{\phi_{\min}^2}.(\phi_i + \phi_{\min})^2 - r_{c\min}, \quad se \quad d_i \le d_{\min}, 0^\circ > \phi_i \ge -\phi_{\min} \; (e) \\ -r_{c\min}, \quad se \quad d_i > d_{\min}, \phi_{\min} > \phi_i > -\phi_{\min} \; (f) \end{cases} \tag{5.16}$$

A Figura 5.10 foi definida de modo a calcular o raio de curvatura quando o robô se encontra à esquerda do alvo, da seguinte maneira:

a) caso o ângulo de inclinação $\phi_i$ esteja abaixo do valor mínimo $-\phi_{\min}$, mas acima de $-90^\circ$, independente da distância $d_i$ normal à reta $r_{ao}$, o raio deve ser o mínimo $-r_{c\min}$ para dar a maior guinada à direita e trazer o robô para a trajetória ótima;

b) caso o robô esteja próximo da trajetória ótima ($d_i \le d_{\min}$) e o ângulo de inclinação esteja acima de $\phi_{\min}$, indicando que o robô está com a sua orientação muito inclinada à direita, é necessário dar uma guinada à esquerda, com $r_{c\min}$;

c) caso o robô esteja mais afastado da trajetória ($d_i > d_{\min}$), mas o ângulo de inclinação esteja acima de $\phi_{\min}$, indicando que o robô está com a sua orientação muito inclinada à direita, é necessário dar uma guinada à esquerda com a metade do raio máximo ($r_{c\max} / 2$);

d) caso o robô esteja próximo da trajetória ótima ($d_i \le d_{\min}$) e o ângulo de inclinação $\phi_i$ esteja entre os valores 0 e $\phi_{\min}$, indicando uma inclinação pequena, o raio é calculado segundo a equação (5.16), item *d* e a Figura 5.10 mostra graficamente este comportamento (curva do primeiro quadrante). O raio é positivo, indicando uma rotação da plataforma do robô para a esquerda;

e) caso o robô esteja próximo da trajetória ótima ($d_i \leq d_{\min}$) e o ângulo de inclinação $\phi_i$ esteja entre os valores $-\phi_{\min}$ e $0$, indicando uma inclinação pequena, o raio é calculado segundo a equação (5.16), item *e* e a Figura 5.10 mostra graficamente este comportamento (curva do terceiro quadrante). O raio é negativo, indicando uma rotação da plataforma do robô para a direita;

f) caso o robô esteja distante e o ângulo de declividade $\phi_i$ esteja entre os valores $-\phi_{\min}$ e $\phi_{\min}$, o raio deve ser o mínimo, $-r_{c\min}$, para dar a maior guinada à direita e trazer o robô para a trajetória ótima.

Duas outras situações de posicionamento do robô podem ocorrer caso o robô encontre obstáculos intransponíveis continuamente, seja pela direita seja pela esquerda, podendo acontecer de a direção do robô ficar contrária à direção do alvo. Estas situações são mostradas na Figura 5.12 e Figura 5.13.

Para a situação apresentada na Figura 5.12, quando $90^\circ \leq \phi_i < 180^\circ$, ou seja, o robô na direção contrária ao alvo e ele se encontrar no lado direito da reta do alvo, tem-se $r_i = -r_{c\min}$. Este caso é ilustrado no instante *i* e *j*. Nos instantes *k*, *l* e *m*, o comportamento é similar ao ilustrado na Figura 5.11.

Para a situação apresentada na Figura 5.13, quando $-90^\circ \geq \phi_i > -180^\circ$, ou seja, o robô na direção contrária ao alvo e ele se encontrar no lado esquerdo da reta do alvo, tem-se $r_i = r_{c\min}$. Este caso é ilustrado no instante *i* e *j*. Nos instantes *k*, *l* e *m*, o comportamento é similar ao ilustrado na Figura 5.9.

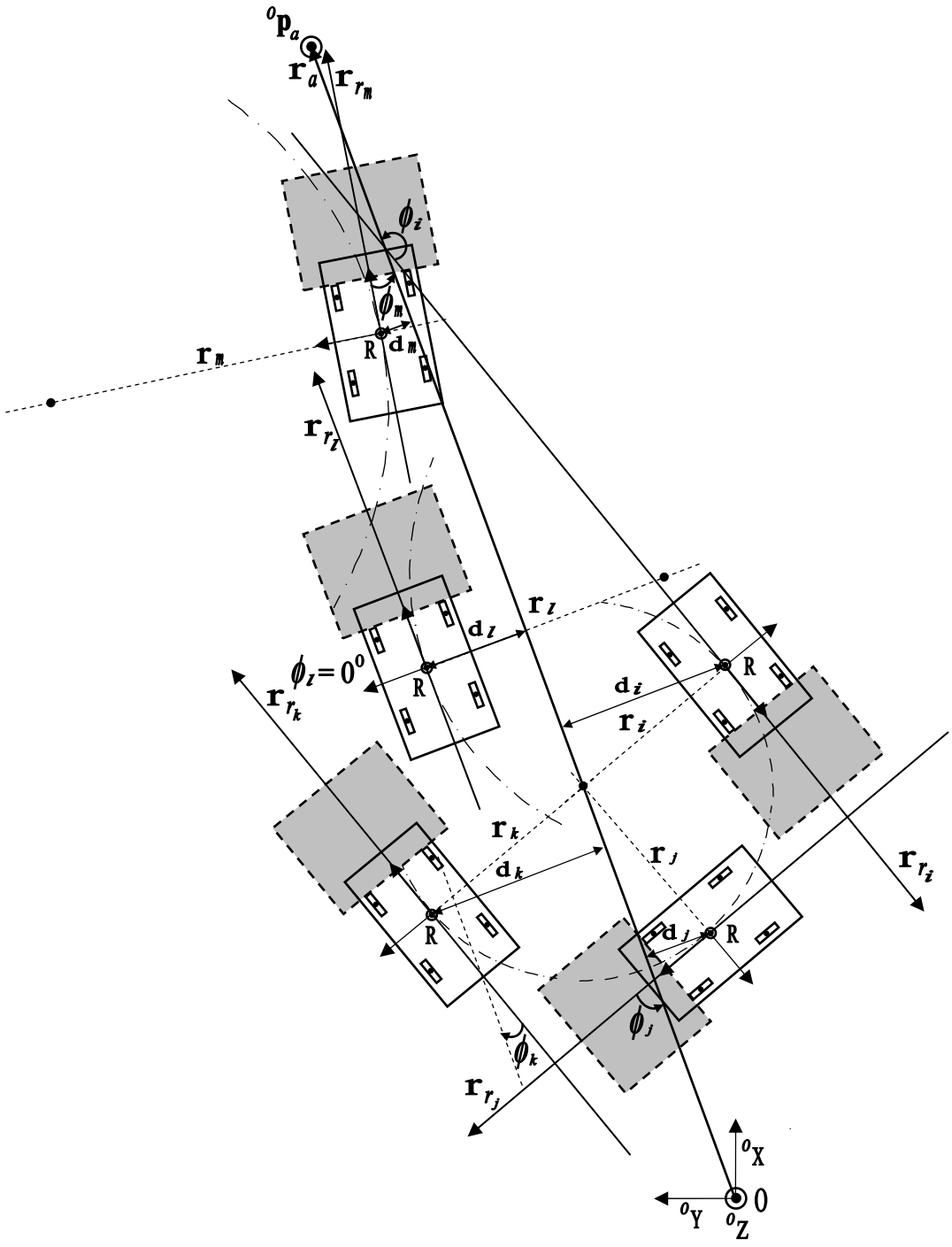


*Figura 5.12: Correção da trajetória na ausência de obstáculos, com robô na direção contrária e à direita do alvo.*

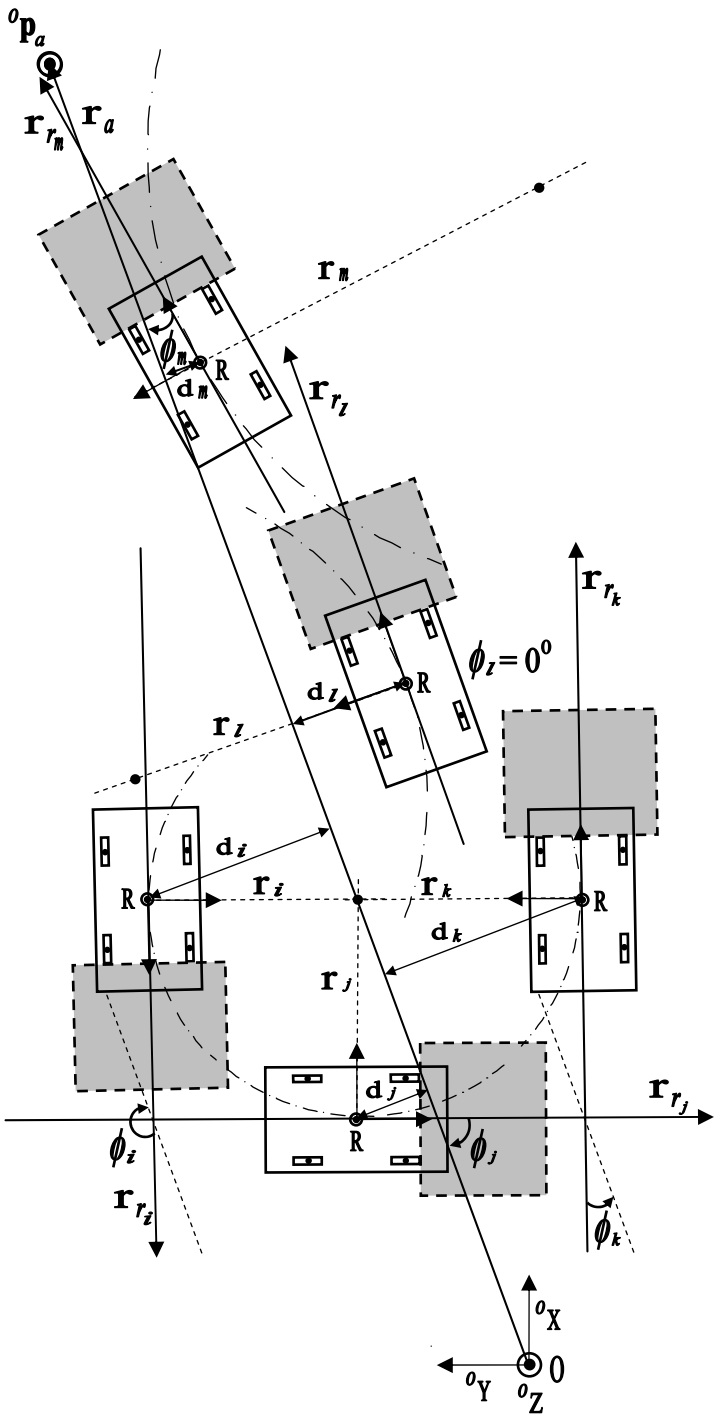


*Figura 5.13: Correção da trajetória na ausência de obstáculos, com robô na direção contrária e à esquerda do alvo.*

## 5.6 Exemplo de Tomada de Decisão

Para descrever um exemplo de tomada de decisão e mostrar o percurso percorrido pela árvore de decisão, a Figura 5.14 ilustra uma situação em que o robô, após caminhar alguns passos em trajetória retilínea rumo ao alvo, depara-se com um obstáculo intransponível à direita, situação retratada pela Figura 5.4.

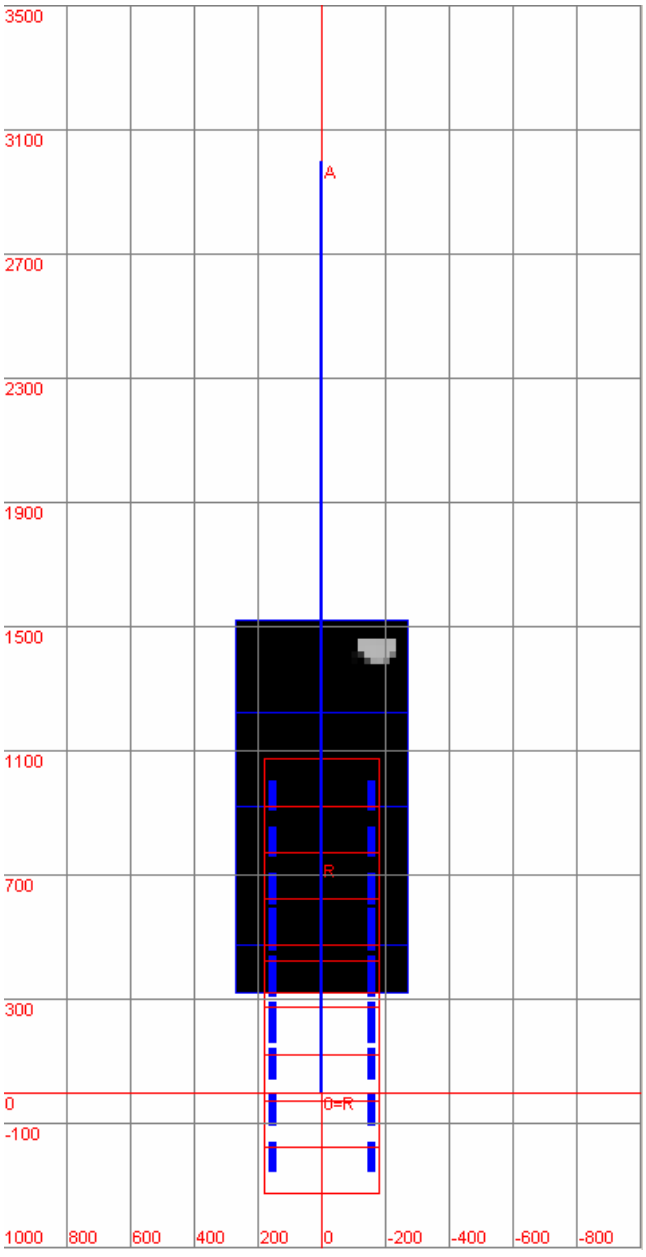


*Figura 5.14:Exemplo de tomada de decisão.*

Ao fazer o mapeamento e constatar a existência do obstáculo, o seguinte percurso da árvore de decisão é feito:

$$\text{C2} \rightarrow \text{C3} \rightarrow \text{C8} \rightarrow \text{C10} \rightarrow \text{C15} \rightarrow \text{C16} \rightarrow \text{A4},$$

e os seguintes dados relevantes são obtidos:

1) obstáculo: centróide ${}^{o}\mathbf{p}_{CO} = \begin{bmatrix} 1444 & -172 & 170 \end{bmatrix}^T$ em mm, altura $h_O = 187$ mm, largura $w_O = 120$ mm, comprimento $l_O = 60$ mm, área = 960 mm$^2$ e volume = 144.040 mm$^3$;
2) ponto para desvio à esquerda, ${}^{o}\mathbf{p}_e = \begin{bmatrix} 1339 & -51 & 0 \end{bmatrix}^T$, em mm;
3) ponto médio, ${}^{o}\mathbf{p}_{e2} = \begin{bmatrix} 1148 & -103 & 0 \end{bmatrix}^T$, em mm;
4) declividades $m_t = 3,673$, $m_r = -0,272$ e $m_s = 0,000$;
5) centro de curvatura, ${}^{o}\mathbf{p}_c = \begin{bmatrix} 750 & 1358 & 0 \end{bmatrix}^T$, em mm;
6) raio de curvatura, $r_C = 1358$ mm;
7) ângulo de guinada, $\phi = 15,23^{\text{o}}$;
8) ação de controle = 4 (desvio à esquerda).

Desta maneira, o raio de curvatura e o ângulo de guinada são enviados para o controle do robô, e o próximo passo é programado para trajetória curvilínea com o raio calculado. Assim, o robô se desviará do obstáculo, de forma ótima, tangenciando o obstáculo, mas mantendo uma distância de guarda segura.

# Capítulo 6 - Resultados Experimentais

Os resultados experimentais do sistema estão separados em 3 grupos:

- ✓ sistema de visão a laser;
- ✓ mapeamento da superfície;
- ✓ sistema de controle de navegação.

## 6.1 Sistema de Visão

O desenvolvimento do sistema de visão proposto foi idealizado para o uso no robô Guará. Os principais recursos do sistema, bem como as principais telas do *software* de controle, são mostrados a seguir.

A Figura 6.1(a) mostra uma imagem do sistema. A placa eletrônica usa o micro processador PIC18F4550 do fabricante *Microchip*, operando com um *clock* de 20 MHz e interface *USB* para comunicação serial com o computador. Nesta placa, há o circuito de controle das fases do motor de passo e as fontes de alimentação do microcontrolador, motor de passo e laser.

A Figura 6.1(b) mostra detalhes da parte inferior do sistema. O canhão do laser utilizado gera um plano de luz, de cor vermelha brilhante, com comprimento de onda de 635 nm, de baixa potência, com 3,5 mW, classe IIa, alimentado em 3 Vcc. O espelho rotativo é de alta reflexão e é fixado ao eixo do motor de passo.

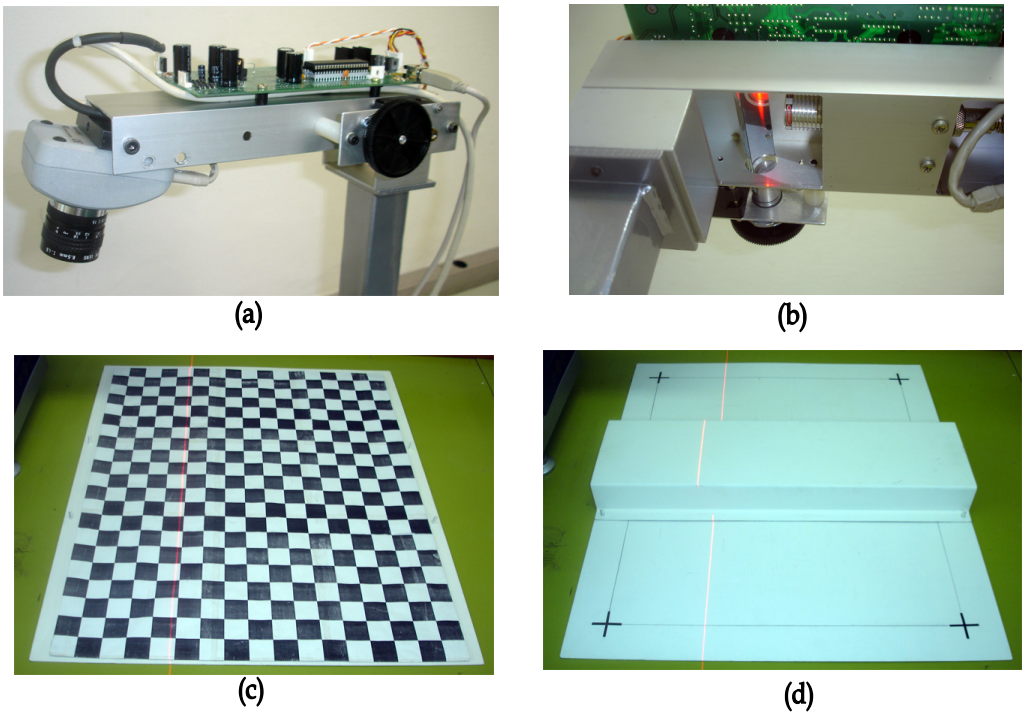


*Figura 6.1: Detalhes do sistema de visão a laser.*

A câmera *CCD* utilizada é do padrão IEEE 1394 (*Firewire*), colorida, configurada para uma resolução de 1280 x 1024 *pixels*. Para a calibração da câmera *CCD*, foi feito um plano quadriculado com marcas pretas e brancas, cada uma de 30x30 $mm^2$ de área. O quadro da imagem (*frame*) foi definido para uma superfície com comprimento de 540 mm e largura de 450 mm, conforme visto na Figura 6.1(c), devidamente mapeado em 640 x 512 *pixels*. A calibração do laser é realizada incidindo-se linhas de laser, para cada passo do motor, do início ao fim do quadro, e armazenando-se as cotas de altura do objeto calibrador. Assim, para cada linha de incidência do laser é associada uma equação da reta. Para tal, um plano com um objeto de altura de 50 mm foi elaborado, conforme visto na Figura 6.1(d).

A Figura 6.2 mostra a tela de calibração da câmera *CCD*. Na parte inferior localiza-se a linha de laser. O algoritmo de correção da distorção radial é executado antes do processo de calibração. As 4 marcas fiduciais para calibração são obtidas através do *click* do *mouse* sobre as cruzes delimitadoras do quadro. Elas mapeiam o plano da imagem em relação ao plano da superfície, usando a transformação projetiva. Os *pixels* não mapeados na transformação projetiva são preenchidos com interpolação bilinear.

Existem as seguintes opções para a imagem corrigida: *distorção radial*, *transformação projetiva* e *interpolação bilinear*. Estas correções podem ser habilitadas ou não com o *click* do *mouse*. Antes da calibração, a opção da *transformação projetiva* deve estar desabilitada.

Após posicionar as 4 marcas fiduciais, basta clicar no botão *Calibra Câmera*, habilitar a opção *transformação projetiva* e verificar que as 4 cruzes se posicionam nos extremos do quadro. Então basta clicar no botão *Salva Calibração* para encerrar o processo. O botão *Carrega Calibração* permite restaurar os dados de calibração gravados em arquivo.

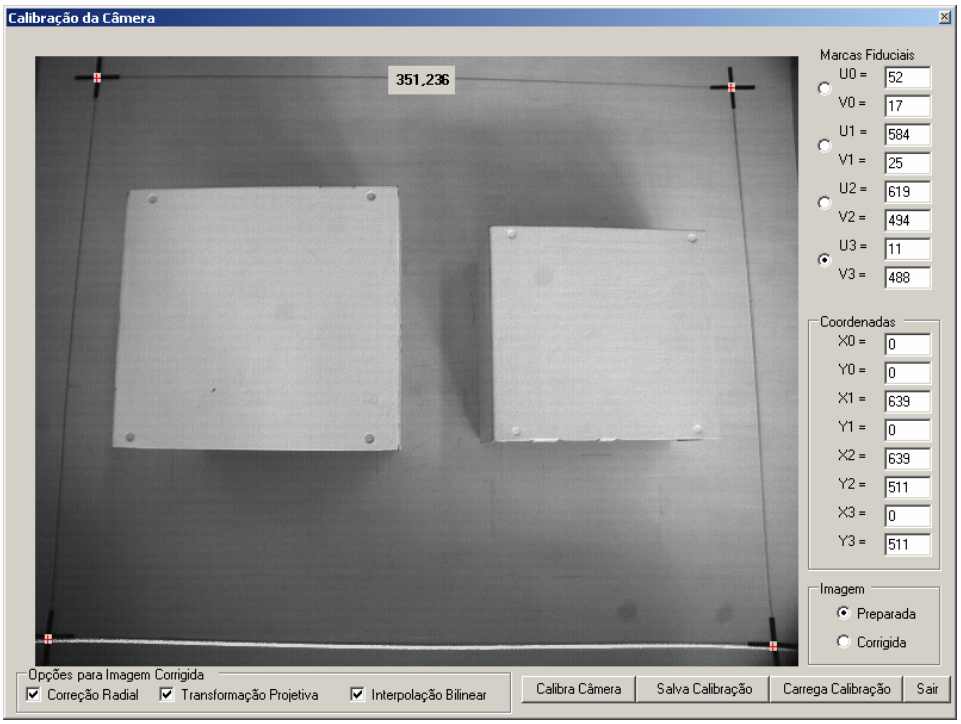


*Figura 6.2: Tela de calibração da câmera CCD antes de calibrar.*

A Figura 6.3 mostra a tela de calibração após calibrar. As quatro marcas fiduciais delimitam os extremos da imagem calibrada. Após a calibração, os dados são salvos em arquivos, e posteriormente estes dados são recuperados, quando o sistema é reiniciado.

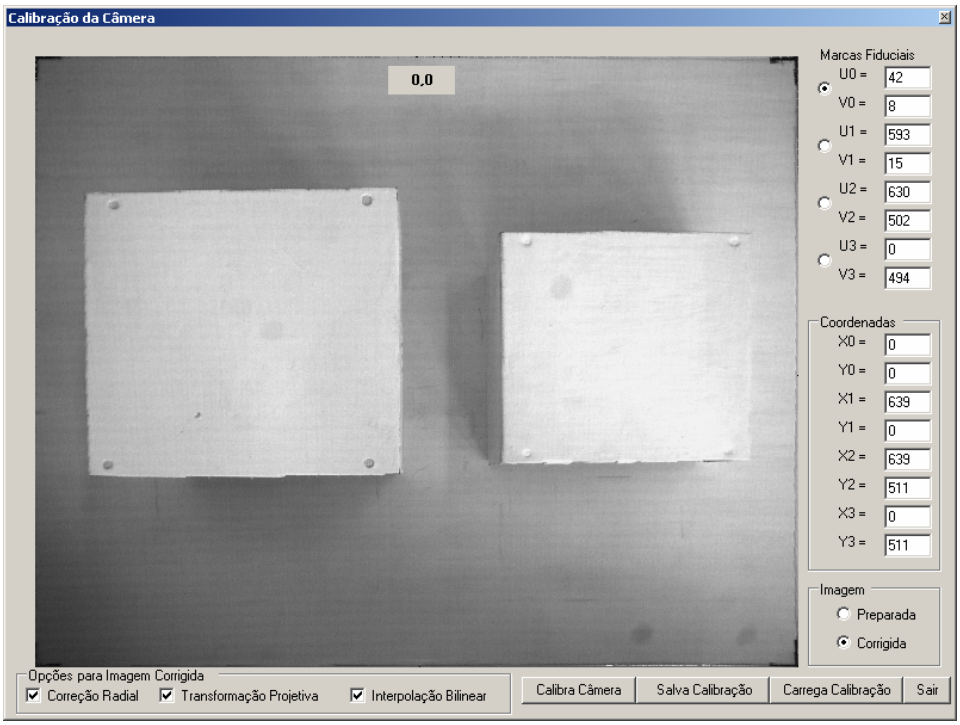


*Figura 6.3: Tela de calibração da câmera CCD após calibrar.*

A calibração do laser é feita com um objeto de largura e altura conhecidas (Figura 6.1(d)). Ele é colocado no meio do plano da imagem, ocupando toda a extensão da largura do quadro. Durante a calibração, o laser varre toda a superfície. A cada linha de laser é associada uma série de parâmetros de calibração. A Figura 6.4 mostra a tela de parâmetros do laser.

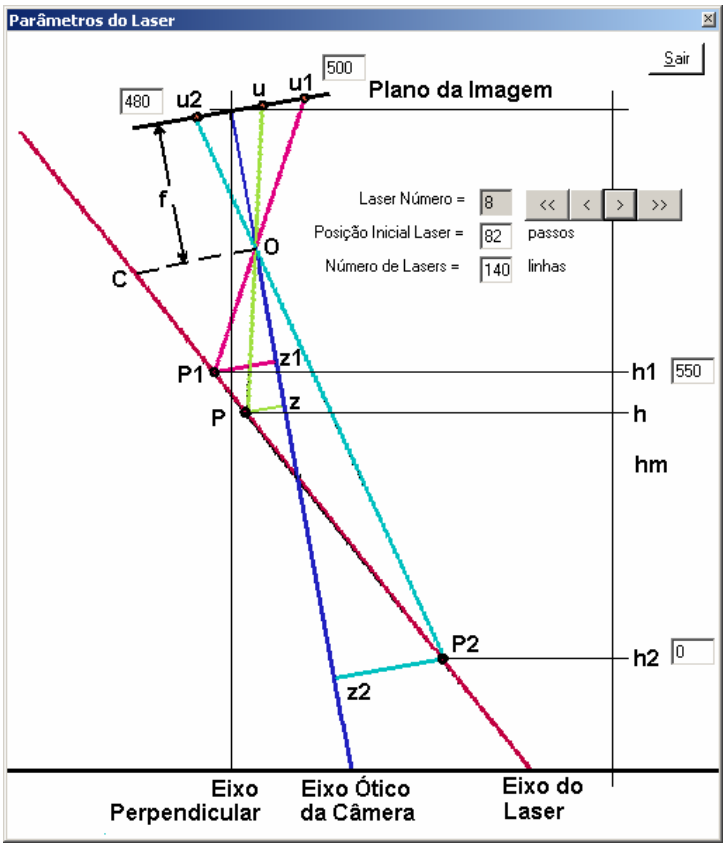


*Figura 6.4: Tela de parâmetros do laser.*

Os resultados experimentais obtidos pelo sistema de visão a laser foram adquiridos com o sistema instalado em uma estrutura fixa, simulando a plataforma do robô Guará.

Vários objetos foram usados para testar o sistema, especialmente objetos com mesma área e com alturas diferentes, para verificação do processo de orto-retificação. O centro ótico da câmera *CCD* ficou a uma altura de 550 mm acima da superfície, sendo equivalente à altura da plataforma do robô.

Para exemplificar os resultados experimentais, a Figura 6.5 mostra a imagem original em tons de cinza, e as suas transformações. Para se obter a imagem em tons de cinza, mostrada na Figura 6.5(a), houve uma sub-amostragem para redução de tamanho original para 640 x 512 *pixels* e uma conversão na imagem colorida original. Na Figura 6.5(b), há o processo de calibração da câmera *CCD*, que corrige a distorção radial e através da transformação projetiva, criando uma relação matemática precisa entre *pixels* e mm. A Figura 6.5(c)

mostra a superfície sendo varrida pelas linhas de laser. Na Figura 6.5(d) a imagem de profundidade é obtida; contudo ela contém a distorção por efeito de perspectiva. A Figura 6.5(e) mostra a imagem de profundidade orto-retificada, para corrigir o efeito da perspectiva. A mesma imagem, em vista *3D*, é mostrada na Figura 6.5(f).

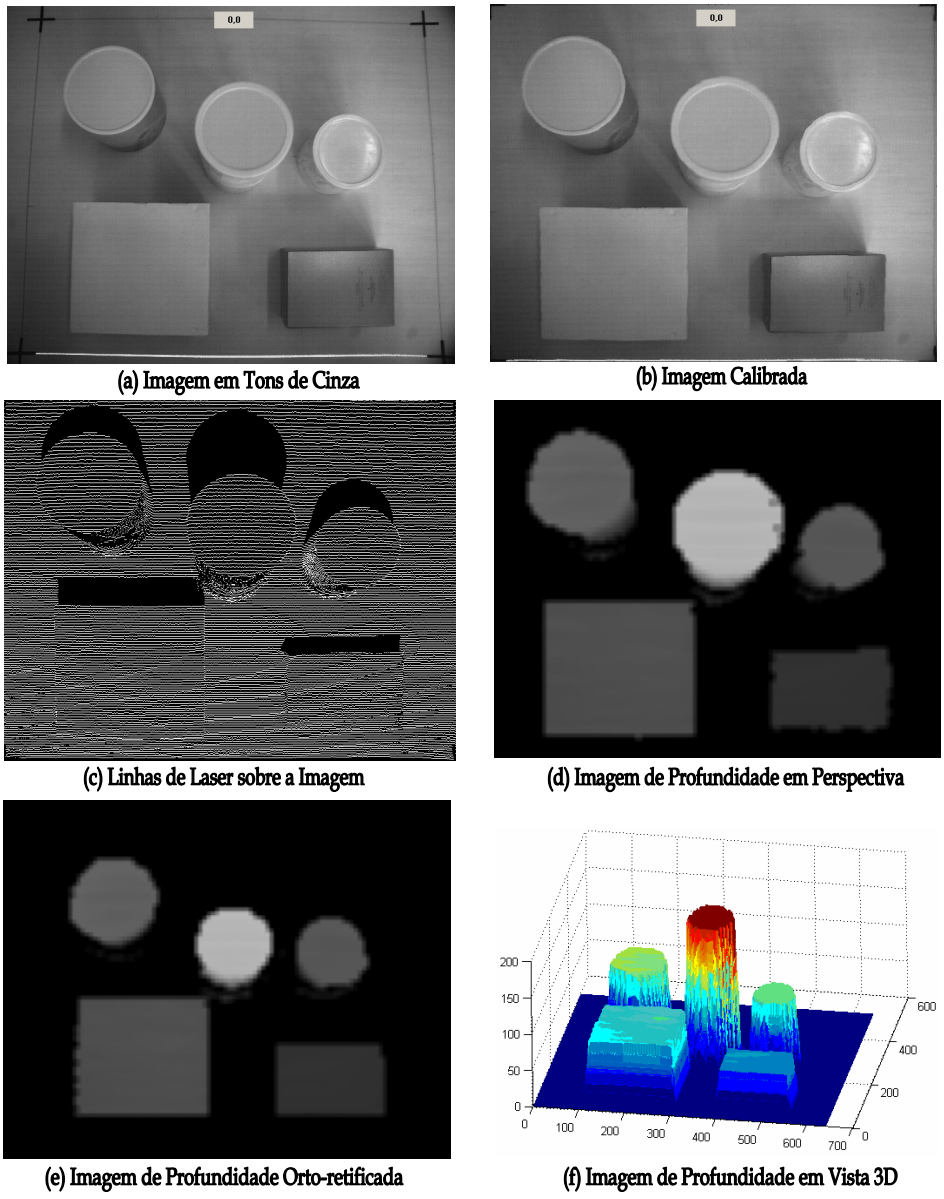


*Figura 6.5: Imagem e as suas transformações.*

Como resultado final, o sistema de visão a laser gera imagens de profundidade precisas, cujo sistema de coordenadas está localizado no canto superior esquerdo do quadro. Estas imagens são a base para o mapeamento da superfície.

Uma análise de erro foi feita para verificar a precisão das medidas. A Figura 6.6 mostra o mapa dos 5 obstáculos mostrados na Figura 6.5 com cotas de distâncias importantes entre eles.

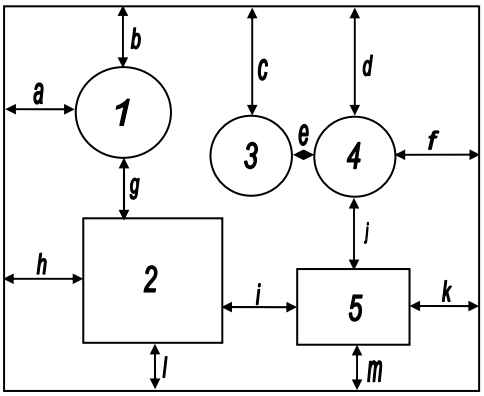


*Figura 6.6: Cotas de distâncias entre os obstáculos.*

A Tabela 6.1 mostra uma comparação entre as distâncias reais e as distâncias medidas pela análise da imagem de profundidade orto-retificada. O erro porcentual é dado em relação à faixa de medição, ou seja, se a cota for vertical, este erro é dado em relação à largura do quadro (450 *mm*) e se a cota for horizontal, o erro é dado em relação ao comprimento (540 *mm*) e se for a elevação, o erro é dado em relação à altura máxima da calibração do laser (250 *mm*) .

Observando a variação do erro, nota-se que as cotas mais distantes, *b*, *c* e *d*, estão com um erro mais significativo. Durante o processo de calibração do laser, as cotas dos perfis de laser no final da imagem têm valor menor em relação à distância total. Conseqüentemente, elas estão mais susceptíveis à interferência da distorção radial e das transformações geométricas que a imagem original sofre.

O robô Guará dá um passo médio completo a cada 150 mm; logo, para que a superfície de análise seja ultrapassada, são necessários 3 passos. As regiões que estão distantes do robô

nos primeiros quadros ficam próximas nos quadros seguintes, aumentando a precisão das medidas nestas novas posições.

*Tabela 6.1: Análise de erro do mapa do quadro.*

| Análise de Erro do Mapa do Quadro | | | |
|---|---|---|---|
| Número do Objeto | Altura Real (mm) | Altura Medida (mm) | Erro (%) |
| 1 | 100 | 98 | -0.80 |
| 2 | 83 | 80 | -1.20 |
| 3 | 180 | 180 | 0.00 |
| 4 | 90 | 88 | -0.80 |
| 5 | 51 | 50 | -0.40 |
| Cotas de Distâncias | Distância Real (mm) | Distância Medida (mm) | Erro (%) |
| a | 80 | 76 | -0.74 |
| b | 85 | 75 | -2.22 |
| c | 155 | 143 | -2.66 |
| d | 155 | 145 | -2.22 |
| e | 37 | 33 | -0.74 |
| f | 105 | 103 | -0.37 |
| g | 65 | 66 | 0.22 |
| h | 87 | 87 | 0.00 |
| i | 67 | 69 | 0.37 |
| j | 85 | 82 | -0.56 |
| k | 83 | 82 | -0.19 |
| l | 50 | 53 | 0.67 |
| m | 50 | 53 | 0.67 |

## 6.2 Mapeamento da Superfície de Navegação

Para mapeamento da superfície, o sistema de visão foi instalado e integrado ao robô Guará, e as medições foram feitas em duas situações:

- ✓ robô com mapeamento manual sobre a trajetória a ser mapeada;
- ✓ robô com mapeamento automático sobre a trajetória a ser mapeada.

O mapeamento manual, denominado *offline*, permite fazer os testes sem que haja a interferência dos erros de odometria do robô. A cada passo do robô, ele pára, e é ajustada a plataforma (translação e rotação), caso necessário, para a trajetória previamente estipulada. Os dados corretos de odometria são inseridos no sistema de visão de forma manual. O algoritmo *ICP* não é acionado neste caso. Desta maneira, a precisão do mapeamento pode ser verificada.

No mapeamento automático, denominado *online*, o robô percorre a trajetória previamente estipulada, e os dados de odometria do robô são enviados para o sistema de visão de forma automática. Nesta situação, há erros de odometria, e o algoritmo *ICP* é acionado para reduzi-los.

Os testes *offline* e *online* foram realizados em uma superfície conhecida, com 2 *m* de largura por 4 *m* de comprimento, onde 4 obstáculos conhecidos são posicionados para o mapeamento.

A Figura 6.7 mostra o mapa real da superfície de testes de mapeamento. Uma vez que o robô é posicionado com centro geométrico no ponto inicial, o sistema de visão a laser é acionado para se fazer a *varredura* sobre a superfície e mapear os obstáculos. A trajetória foi definida para que o robô passasse sobre os obstáculos, exceto o último, que simula um degrau de uma escada, onde ele pára.

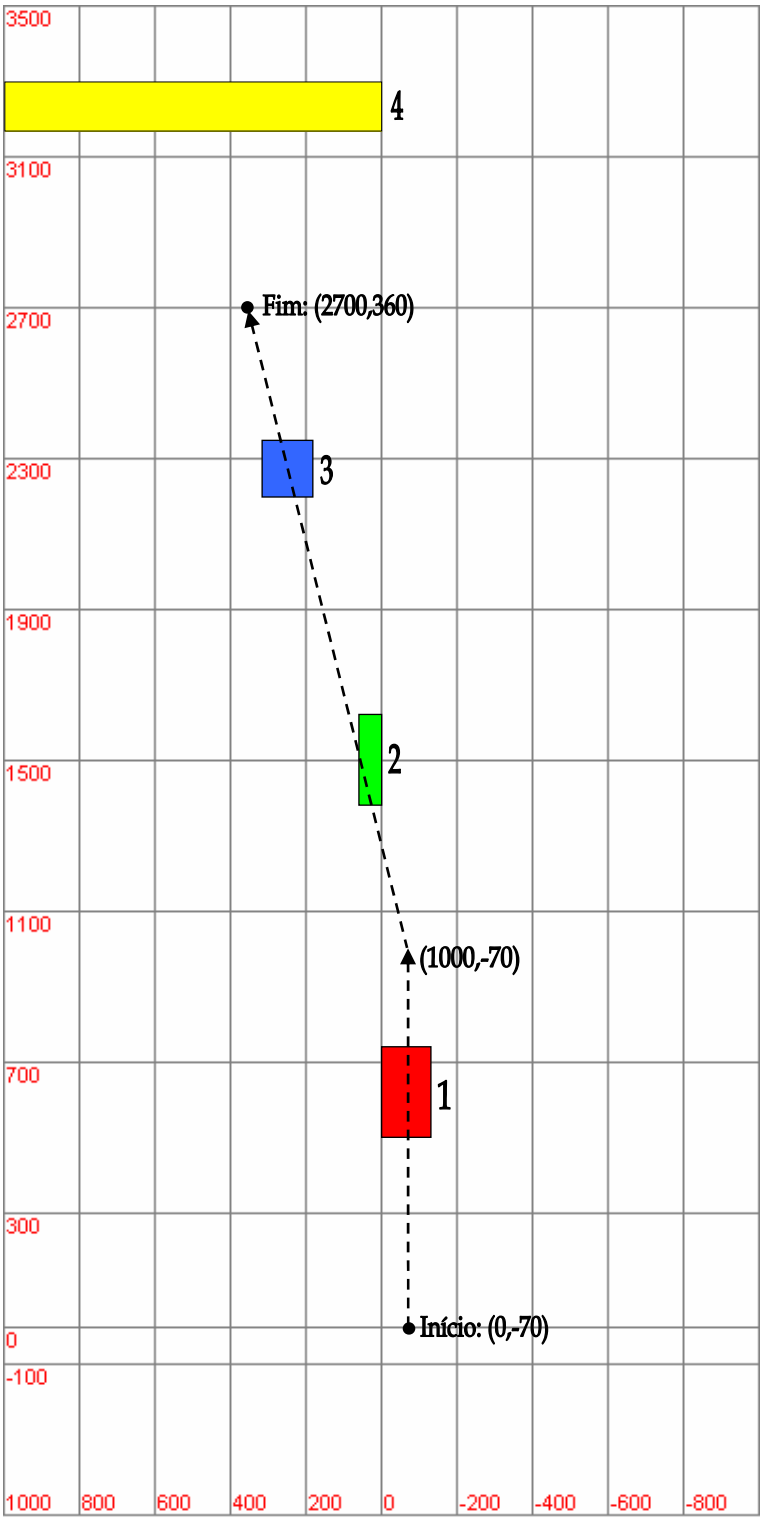


*Figura 6.7: Mapa da superfície com os obstáculos (em mm).*

Nesta trajetória, o robô anda por 1 *m* para frente em linha reta, dá uma rotação de 14º para a esquerda, anda em linha reta por 1,75 *m*, dá uma rotação de 14º para a direita e pára, próximo a um degrau.

O robô mede a largura e comprimento do obstáculo em relação à orientação do seu referencial à bordo, que pode ou não estar na mesma orientação dos eixos do referencial do mundo. Para cada obstáculo são medidos o seu centróide, a largura, a altura e o comprimento, para comparação posterior com os dados do mapeamento real.

O passo médio do robô, para testes *offline*, foi definido como 150 mm. Para aumentar a precisão, foi usado o modo de varredura fina do laser, com 140 linhas de laser por quadro de imagem. As imagens de profundidade geradas são então armazenadas. Com os dados de posição do centro geométrico do robô e de sua orientação, é feito o registro de imagens, neste caso sem o uso do algoritmo *ICP*, pois as medidas de posicionamento e orientação são reais.

O passo seguinte é a cubificação das imagens de profundidade e a geração dos descritores dos objetos. Para aumentar a precisão, a resolução da cubificação foi mudada dos atuais 20x20 $mm^2$, para 8x8 $mm^2$ aumentando significativamente o processamento. Trata-se apenas de um ensaio e este valor não é recomendado para o funcionamento normal do robô.

A Figura 6.8 mostra o painel com todas as 20 imagens de profundidade adquiridas pelo mapeamento *offline*, indicando a posição do centro geométrico do robô no plano cartesiano, para cada quadro e quando há a rotação na sua plataforma.

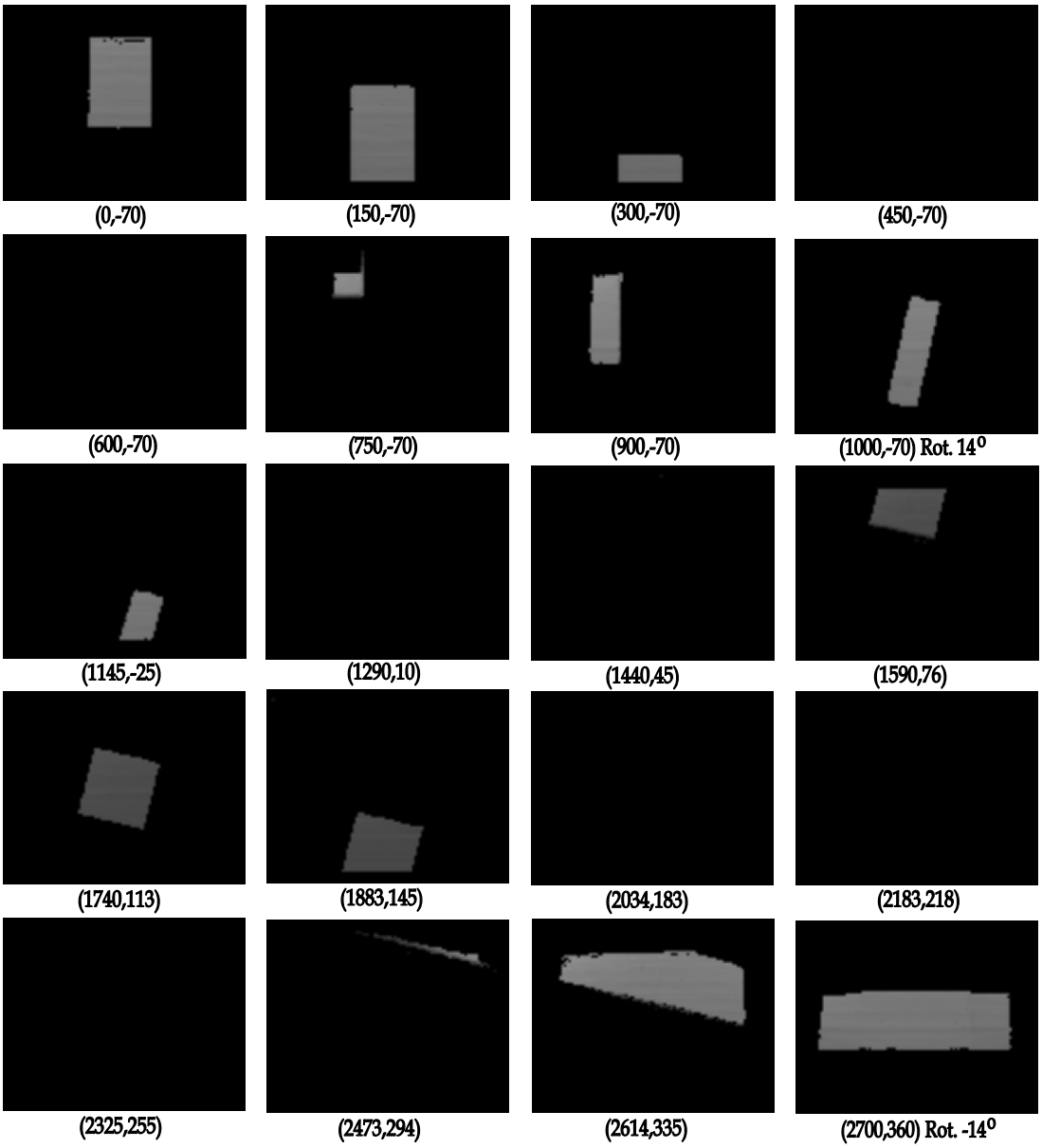


*Figura 6.8: Imagens de profundidade adquiridas no mapeamento offline da superfície pelo robô Guará.*

A Figura 6.9 mostra o mapeamento *offline* realizado pelo sistema de visão. Foram dados 19 passos e obtidas 20 imagens de profundidade, desde o início da trajetória até o ponto de parada, próximo ao degrau.

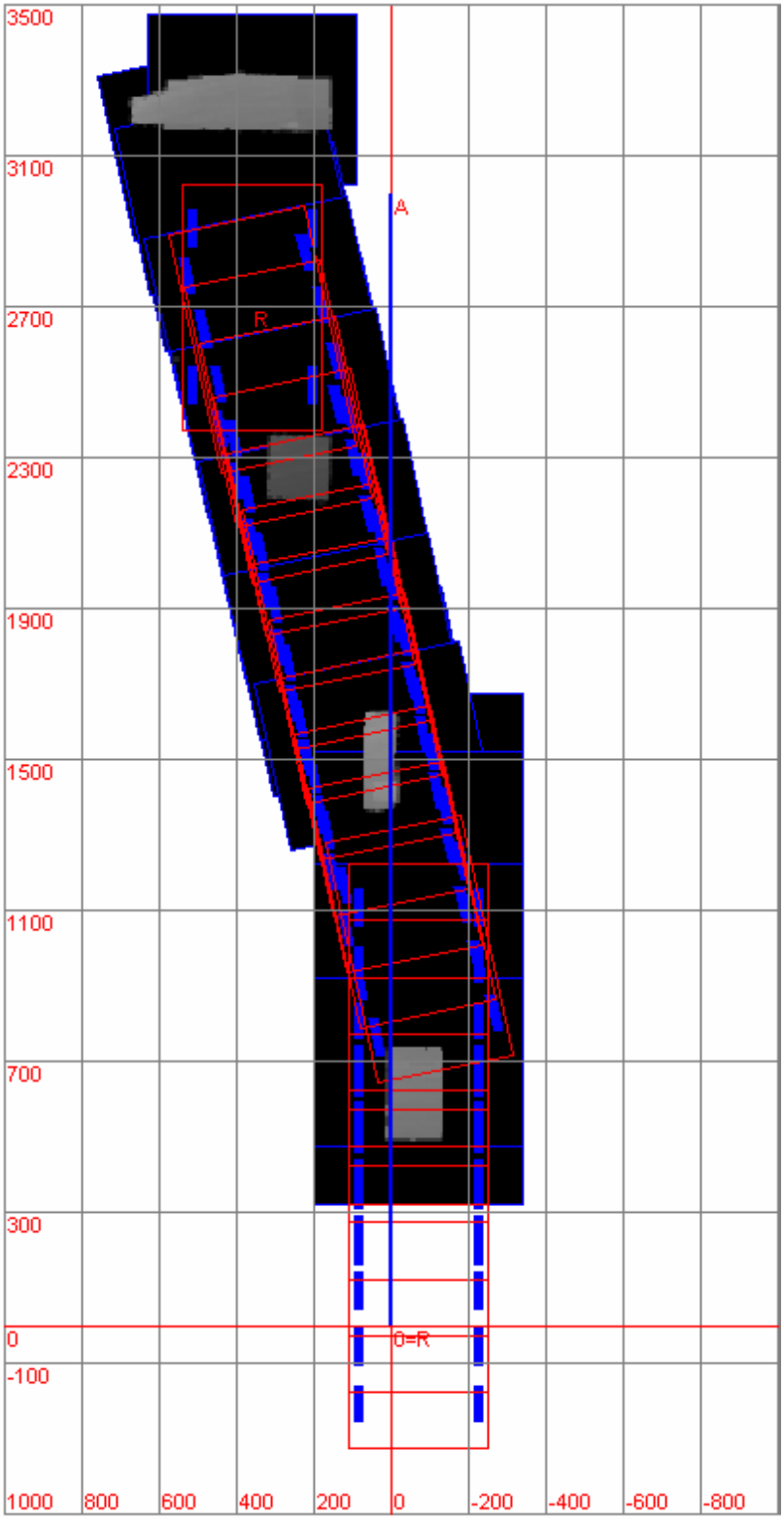


*Figura 6.9: Trajetória e Mapeamento offline da superfície pelo robô Guará (em mm).*

A Figura 6.10(a) mostra o mapeamento *offline* da superfície, com os obstáculos cubificados em tons de cinza e a Figura 6.10(b) mostra os obstáculos isolados e identificados.

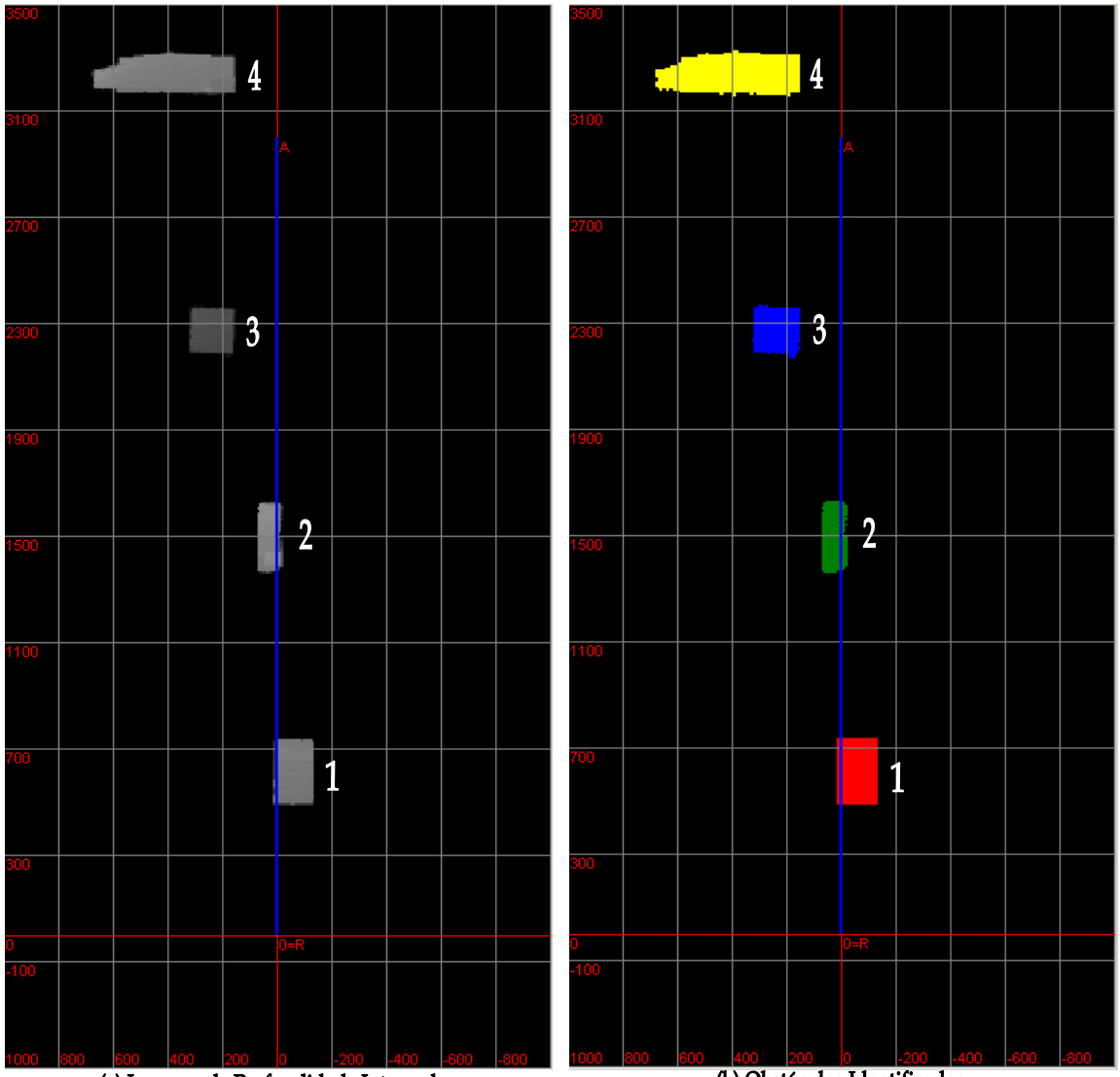


*Figura 6.10: Superfície mapeada offline e obstáculos identificados (em mm).*

A Tabela 6.2 mostra uma comparação das medidas reais dos obstáculos da superfície com os dados obtidos dos descritores dos obstáculos durante o mapeamento *offline*. Diferenças na largura e comprimento são devido ao fato de que estes dois descritores são obtidos em função da orientação da plataforma do robô. Os centróides são mostrados no plano cartesiano.

*Tabela 6.2: Comparação do mapeamento real e medido em offline.*

| Análise de Comparação das Medidas reais com o Mapeamento *offline* da Superfície | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Obs.** | **Valor Real (mm)** | | | | **Valor Medido (mm)** | | | |
| | **Centróide** | **Altura** | **Largura** | **Comp.** | **Centróide** | **Altura** | **Largura** | **Comp.** |
| 1 | (620,-70) | 120 | 130 | 240 | (614,-60) | 117 | 144 | 248 |
| 2 | (1500,30) | 130 | 60 | 240 | (1493,30) | 129 | 123 | 259 |
| 3 | (2270,248) | 83 | 135 | 150 | (2276,240) | 86 | 189 | 197 |
| 4 | (3235,500) | 120 | 1000 | 130 | (3238,401) | 120 | 520 | 159 |

Analisando a Tabela 6.2, exceto o obstáculo 4, que se trata de um degrau e o sistema não o mapeou todo, os centróides dos obstáculos 1, 2 e 3 estão bastante próximos dos valores reais. Para a altura, que não muda com a orientação do robô, nos quatro obstáculos, o erro foi pequeno. Para o obstáculo 1, cuja orientação da plataforma do robô é a mesma que a dos eixos do referencial do mundo, os valores de largura e comprimento estão mais próximos.

O processo de cubificação tende a aumentar os obstáculos, justificando o aumento nestas medidas. Quanto aos obstáculos 2 e 3, a orientação do robô difere da orientação do referencial do mundo e, portanto, os descritores largura e comprimento nestes casos estão diferentes, pois eles foram analisados na inclinação do robô.

Os testes de mapeamento *online* foram baseados em fazer o robô andar pela superfície conhecida de forma automática. Neste caso, o passo médio do robô pode variar em torno de 150 *mm*. A cada passo completo, os dados de odometria do robô são enviados para o sistema de visão via conexão *socket*, usando como meio físico a rede *ethernet*, com o protocolo *TCP/IP*.

Para haver um melhor registro de imagens, o algoritmo *ICP* foi acionado, e a matriz de correção do *ICP* é enviada para o sistema de controle do robô via conexão *socket*. O algoritmo *ICP* só gera a matriz de correção quando há duas imagens sucessivas com áreas

em comum. Desta maneira, o robô corrige a matriz de transformação do passo atual do robô, e os erros de odometria são reduzidos para o próximo passo.

Para poder haver uma comparação com o mapeamento *offline*, a resolução da cubificação de 8x8 $mm^2$ e a varredura do laser no modo fino foram mantidas. Estes dois parâmetros melhoram a precisão das medidas, mas aumentam o tempo de processamento, não sendo recomendados em operação normal.

A Figura 6.11 mostra o momento em que o robô Guará está mapeando a superfície. O sistema de visão está fazendo uma aquisição do quadro.

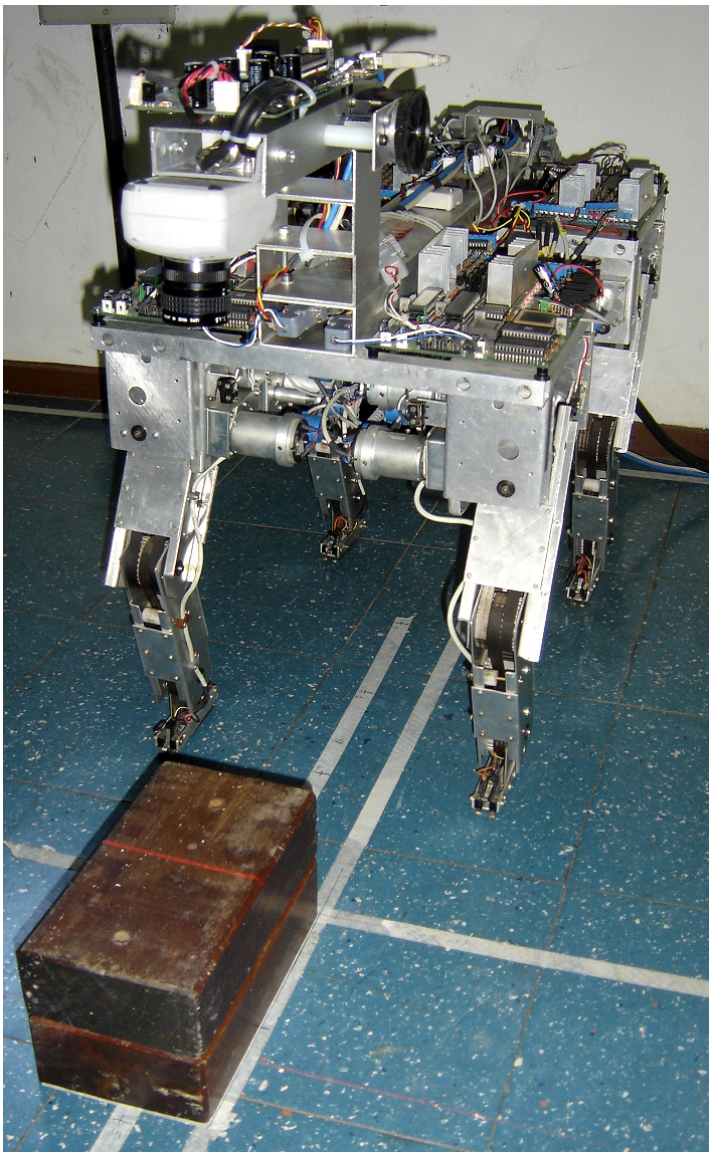

*Figura 6.11: O robô Guará mapeando a superfície.*

Para gerar o mapeamento, o robô andou 19 passos completos, em uma superfície com obstáculos colocados no caminho. A Figura 6.12 mostra as 20 imagens de profundidade adquiridas, e a posição do centro geométrico do robô, no plano cartesiano.

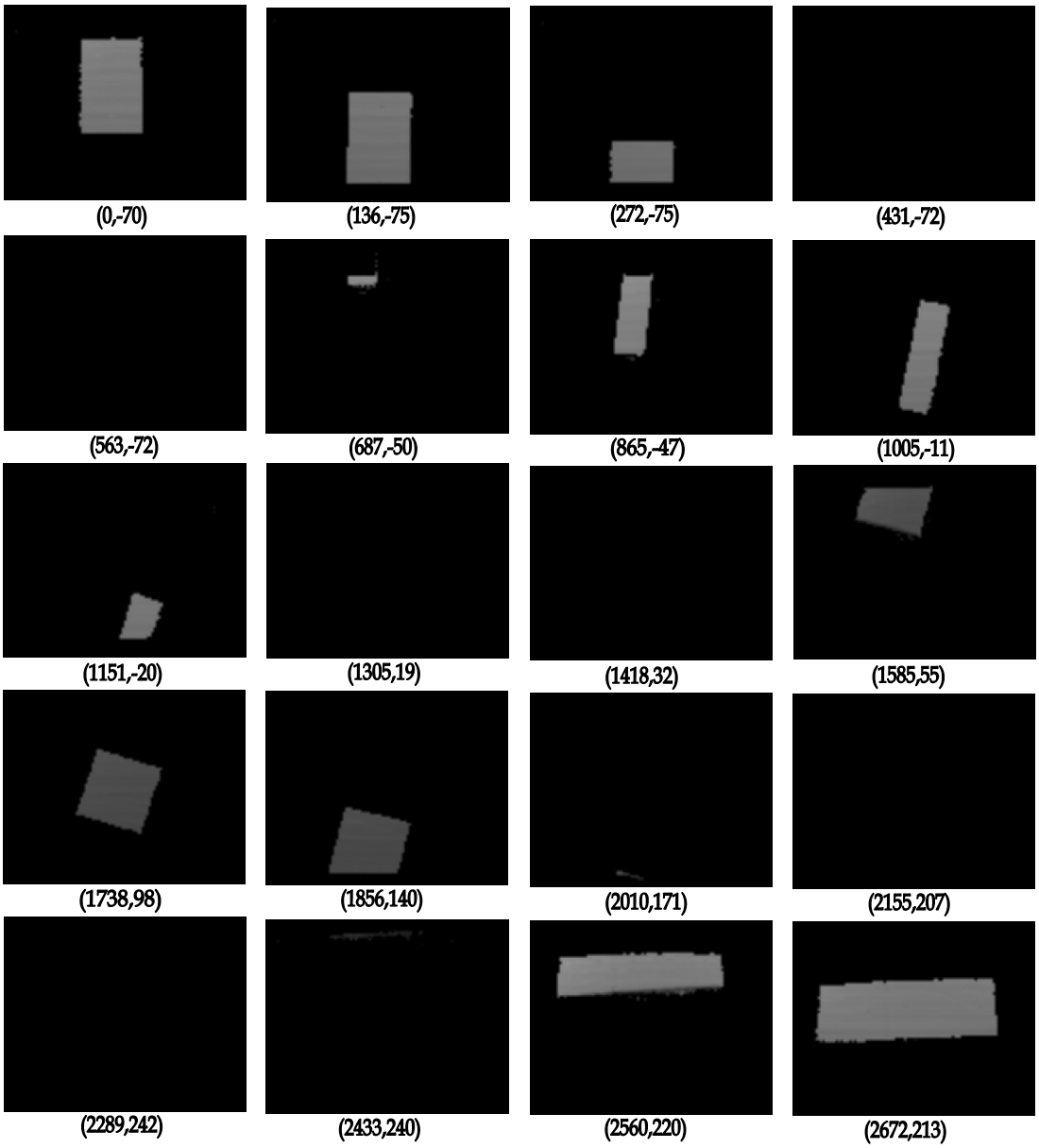


*Figura 6.12: Imagens de profundidade adquiridas no mapeamento online da superfície pelo robô Guará.*

A Figura 6.13 mostra o mapeamento *online* feito pelo robô Guará. A trajetória difere um pouco em relação ao mapeamento *offline*, porque o robô, para fazer curvas, necessita de mais de um passo e por causa dos erros de odometria.

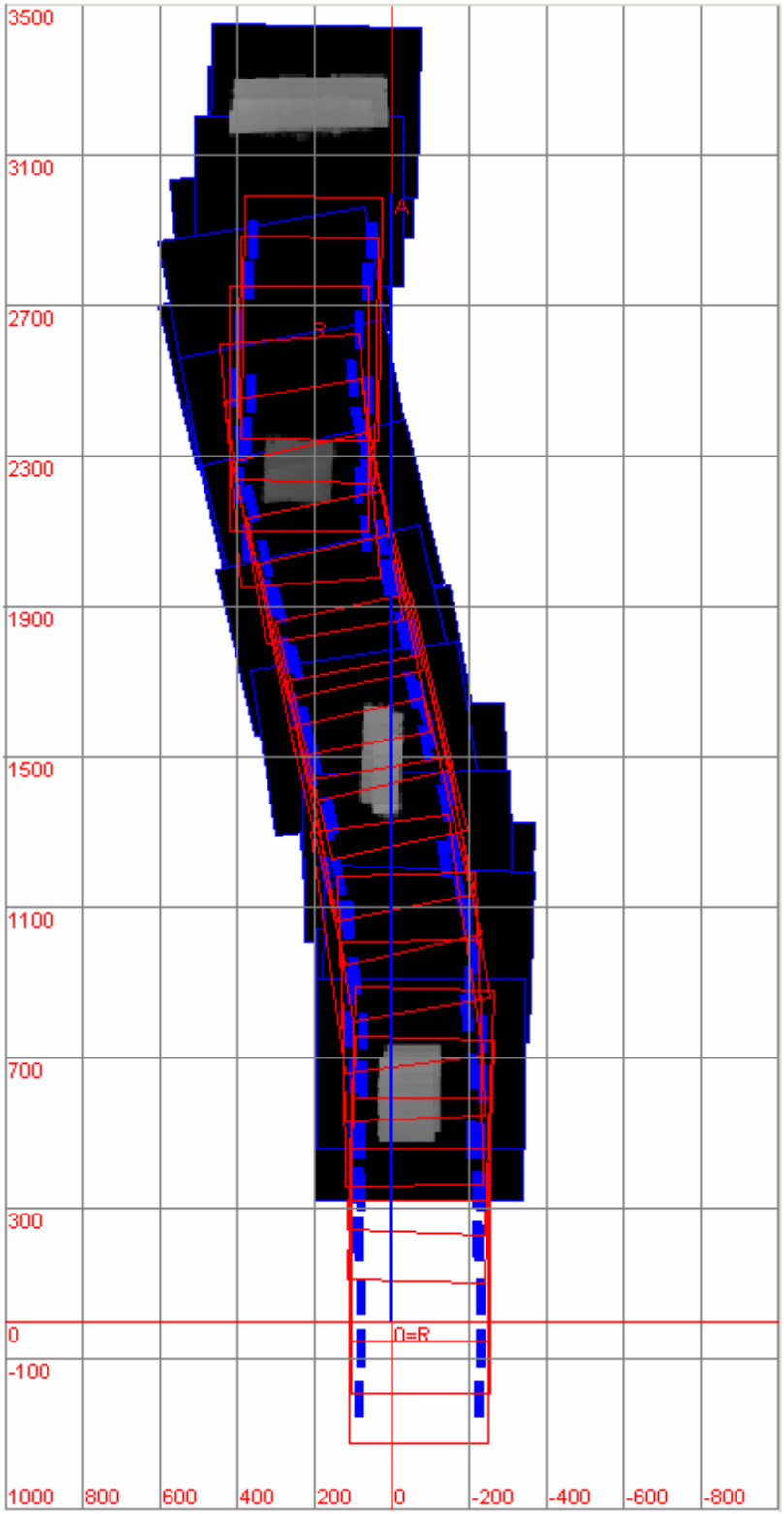


*Figura 6.13: Trajetória e Mapeamento online da superfície pelo robô Guará (em mm).*

A Figura 6.14(a) mostra o mapeamento *online* realizado, enquanto que a Figura 6.14(b) mostra os obstáculos isolados e identificados, cada qual com uma cor diferente associada ao seu número de identificação.

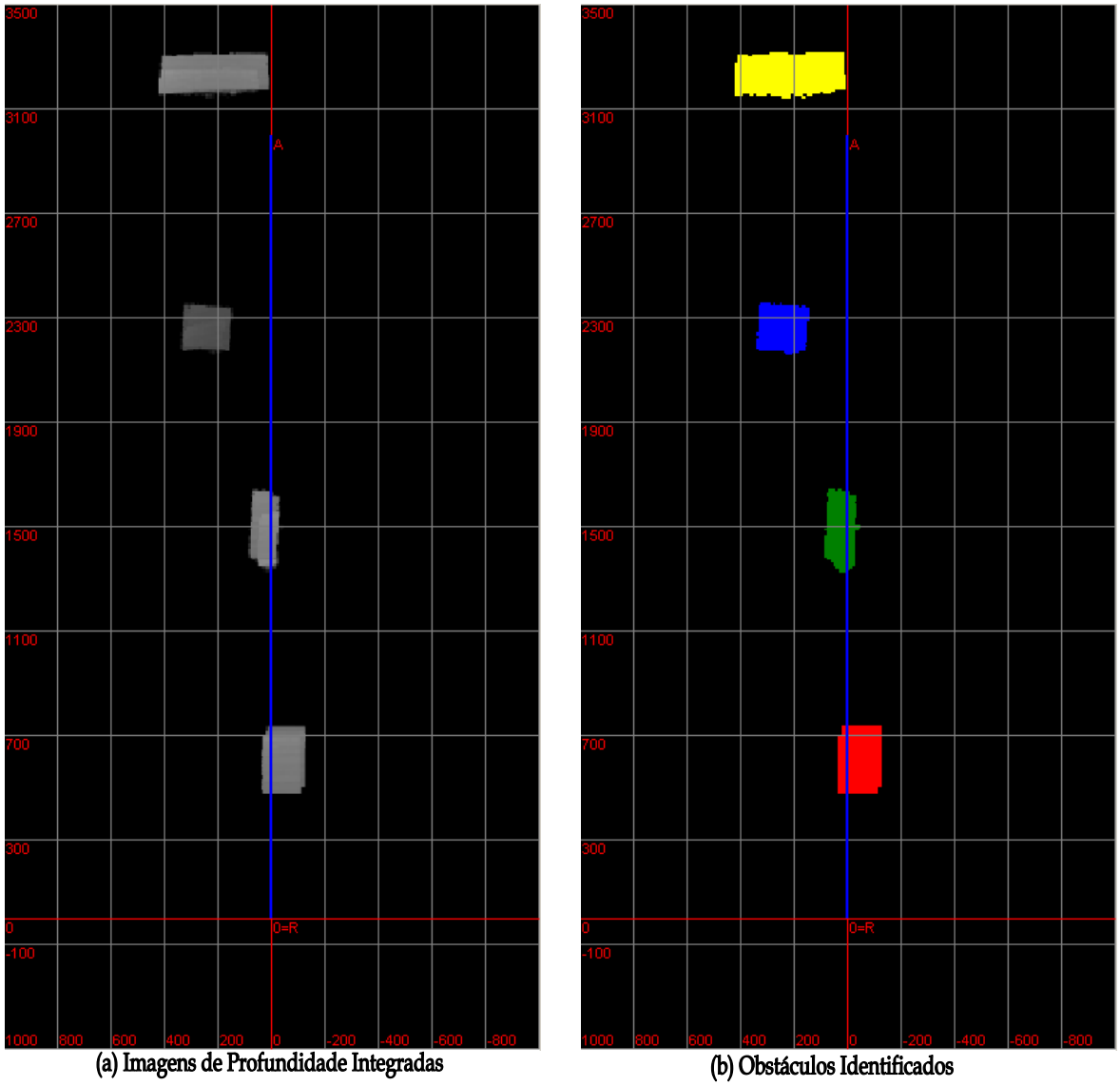


*Figura 6.14: Superfície mapeada online e obstáculos identificados (em mm).*

A Tabela 6.3 mostra uma comparação das medidas reais dos obstáculos da superfície com os dados obtidos dos descritores dos obstáculos durante o mapeamento *online*. Diferenças na largura e comprimento são devido ao fato de que estes dois descritores são obtidos em

função da orientação da plataforma do robô. Os centróides são mostrados no plano cartesiano.

*Tabela 6.3: Comparação do mapeamento real e medido em online.*

| Análise de Comparação das Medidas reais com o Mapeamento *online* da Superfície | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Obs.** | **Valor Real (mm)** | | | | **Valor Medido (mm)** | | | |
| | **Centróide** | **Altura** | **Largura** | **Comp.** | **Centróide** | **Altura** | **Largura** | **Comp.** |
| 1 | (620,-70) | 120 | 130 | 240 | (607,-46) | 117 | 169 | 265 |
| 2 | (1500,30) | 130 | 60 | 240 | (1493,25) | 124 | 136 | 260 |
| 3 | (2270,248) | 83 | 135 | 150 | (2261,244) | 89 | 153 | 205 |
| 4 | (3235,500) | 120 | 1000 | 130 | (3180,214) | 125 | 431 | 159 |

Analisando a Tabela 6.3, observa-se um aumento do erro entre os valores reais e medidos. Os erros de odometria do robô justificam o aumento do erro, que é atenuado com o uso do algoritmo *ICP*. Houve um deslocamento maior dos centróides dos obstáculos. O descritor altura não se alterou muito em ambos os casos, mas a largura e o comprimento aumentaram em conseqüência do erro maior do alinhamento das imagens.

O conjunto de dados descritores do obstáculo é gerado antes da tomada de decisão porque alguns descritores como largura e comprimento dependem da orientação do robô perante o obstáculo.

A Figura 6.15 mostra a tela de *Distâncias para Controle*. Esta tela gera 20 distâncias e 25 pontos mapeados pelo sistema para fornecer os dados da superfície e obstáculos que estão próximos ao robô, dentro dos círculos de procura das patas.

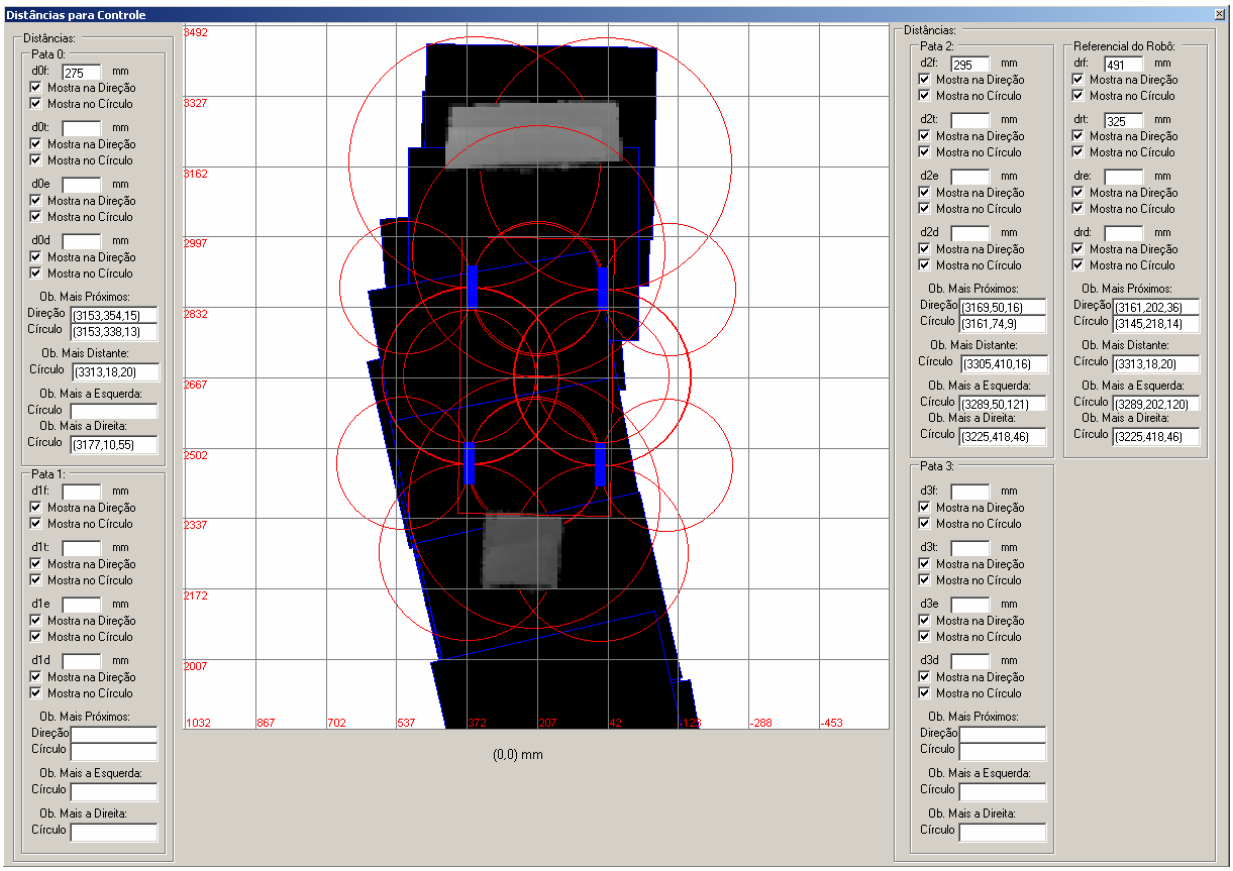


*Figura 6.15: Distâncias e pontos a obstáculos próximos ao robô Guará.*

Cada pata e o centro geométrico do robô têm as seguintes distâncias em relação aos obstáculos: frontal, lateral esquerda, lateral direita e traseira. Estas distâncias são medidas em relação à orientação do sistema de coordenadas do referencial do robô. Os pontos estratégicos de cada pata e centro geométrico são: ponto do obstáculo mais distante, ponto do obstáculo mais próximo, ponto do obstáculo mais à esquerda e , ponto do obstáculo mais à direita, que são obtidos em coordenadas do mundo.

Pelos dados da Figura 6.15, o obstáculo 4 (degrau) encontra-se a 275 mm da pata 0, a 295 mm da pata 2 e a 491 mm do centro geométrico do robô. O obstáculo 3, que o robô passou, encontra-se a 325 mm do centro geométrico.

Os círculos de procura têm a função de otimizar a identificação dos obstáculos. Somente os obstáculos que se encontram dentro ou tocando os círculos são processados, reduzindo o

tempo de processamento. Para cada pata e para o centro geométrico, há 4 círculos de procura: frontal, traseiro, lateral direito e lateral esquerdo.

O mapeamento da superfície, a identificação dos obstáculos e a geração das distâncias e pontos estratégicos das patas e centro geométrico formam um conjunto rico de dados que permite a total identificação do ambiente pelo robô Guará, garantindo uma navegação segura.

## 6.3 Sistema de Controle da Navegação

Os resultados do sistema de controle foram obtidos em 2 situações:

- ✓ robô encontra obstáculo transponível pela plataforma e mantém a trajetória;
- ✓ robô encontra obstáculo intransponível à direita e desvia pela esquerda.

A Figura 6.16 mostra a interface de controle. Esta figura contém todos os referenciais e seus valores atuais, o gráfico da trajetória, dados de comunicação com o sistema do robô, além das matrizes de transformação entre os referenciais do sistema e a matriz de correção do algoritmo *ICP*.

A ação de controle e raio de curvatura para o próximo passo são mostrados. O raio de curvatura é enviado para o sistema de controle do robô para a execução do próximo passo. Neste exemplo, o obstáculo encontrado é classificado como intransponível devido à sua altura, o robô deve executar uma trajetória curvilínea no próximo passo com raio de 1358 mm, cuja ação de controle é A4 (desvio à esquerda), destacada em vermelho.

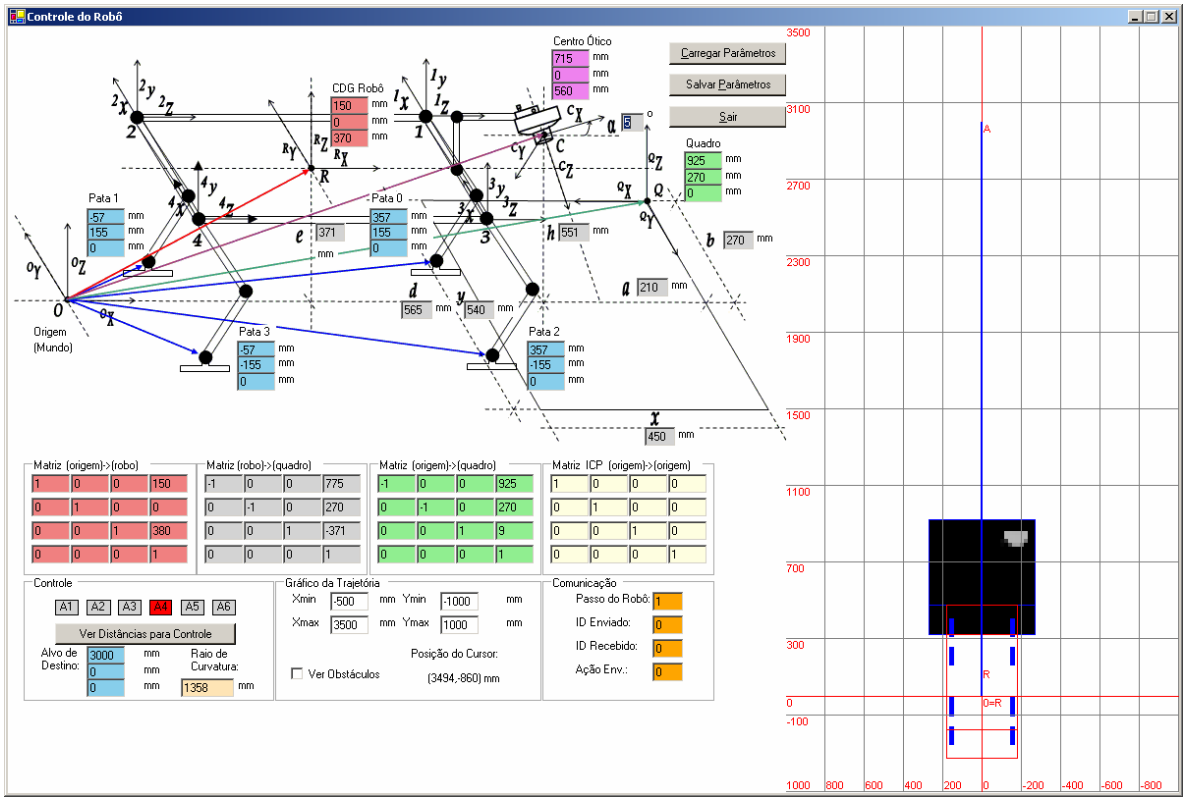


*Figura 6.16: Interface do sistema de controle de tomada de decisão.*

A trajetória mostrada na Figura 6.17 foi gerada quando o robô encontrou um obstáculo transponível entre as patas e manteve a trajetória retilínea rumo ao alvo. Foram registrados 10 passos completos do robô. Ao gerar o obstáculo e analisá-lo, o sistema de controle constatou que suas dimensões estavam de acordo para a transposição pela plataforma, e o robô continuou em trajetória retilínea.

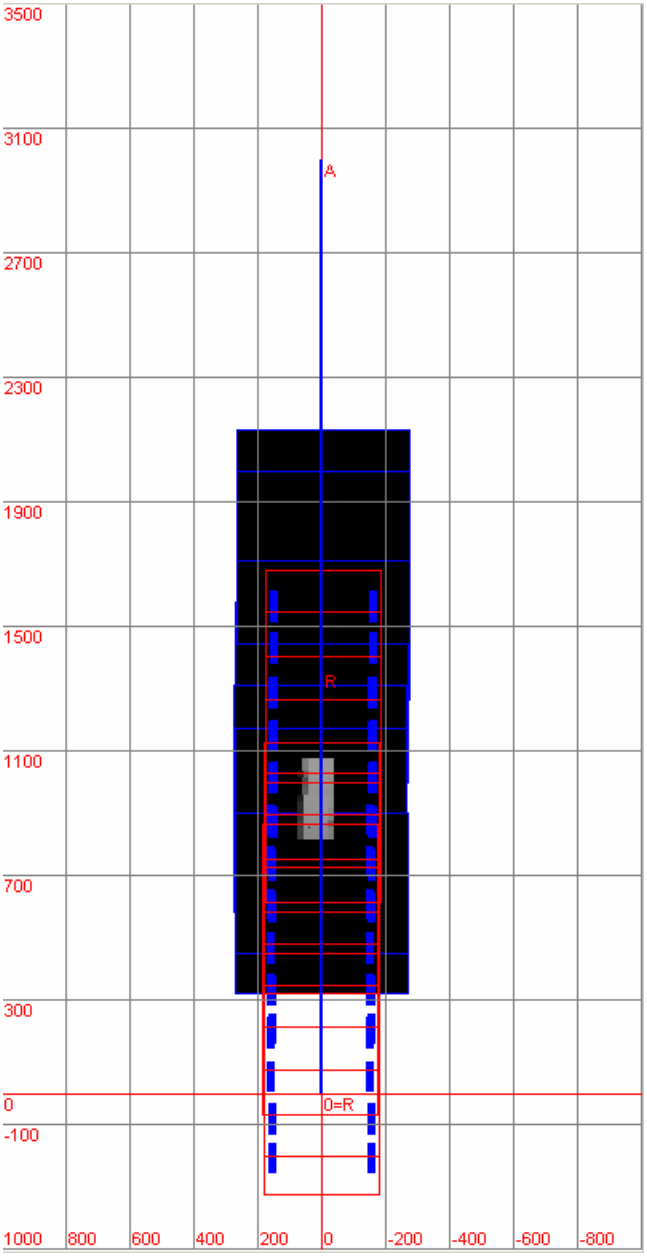


*Figura 6.17: Robô encontra obstáculo transponível.*

A trajetória mostrada na Figura 6.18 foi gerada quando o robô encontrou um obstáculo intransponível à direita. Este obstáculo é alto o suficiente para não ser transpassado pela plataforma.

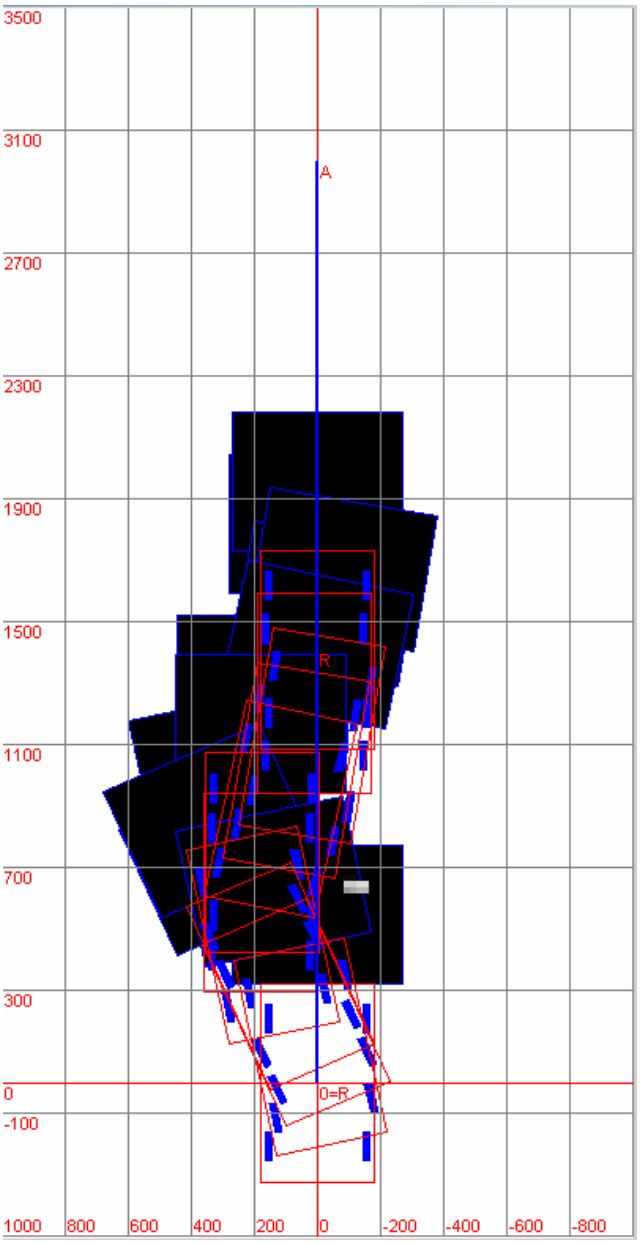


*Figura 6.18: Robô encontra obstáculo intransponível à direita.*

A ação de controle do robô calcula o raio de curvatura para desvio pela esquerda. O robô se desvia do obstáculo dando passos para a esquerda, logo o obstáculo sai fora do campo de visão do sistema, o robô passa a andar em trajetória retilínea, contudo com o afastamento de sua trajetória ótima, o sistema de controle detecta e programa o robô para dar passos para trás, ou seja, para a direita. Quanto o robô se aproxima novamente da trajetória ótima,

o robô é programado para andar na trajetória retilínea, em direção ao alvo. Ao todo, o robô deu 11 passos para fazer esta manobra.

Durante os ensaios, foram constatadas algumas limitações de operação do robô Guará. Os erros de odometria, normais para os robôs, são mais significativos em um robô quadrúpede, devido ao modelo complexo da geração da matriz de rotação e do vetor de translação do centro geométrico do robô, e também devido à quantidade de dados medidos pelas 4 pernas, cada qual com 4 juntas.

As medições das juntas têm incertezas que alteram os dados odométricos do robô. Os atuadores das pernas do robô têm limites cinemáticos restritos, que impedem passos maiores e restringem as curvas com raios de curvatura menores que 1,2 *m*. A elevação da pata em vôo é limitada a 50 *mm*, impedindo que o robô, com estes ajustes atuais, possa passar com a pata sobre obstáculos. Desta maneira, o robô anda de forma lenta.

Quanto ocorre de o robô se deparar com um obstáculo alto e largo e não puder fazer a curva, o controle toma a decisão de dar um passo em marcha à ré e re-analisar o posicionamento do robô.

O mapeamento e a geração de distâncias e pontos estratégicos mostraram-se eficientes e precisos no caso do mapeamento *offline* em que os erros de odometria foram suprimidos. No caso do mapeamento *online*, embora houvesse erros de odometria, o algoritmo *ICP* ajudou a minimizá-los.

# Capítulo 7 - Conclusões

## 7.1 Conclusões

A escolha da técnica de medição de distância do sistema de visão, desenvolvido para o robo quadrúpe Guará, foi definida através da análise dos prós e dos contras de cada técnica estudada. A arquitetura escolhida, cujo princípio baseia-se em triangulação ativa usando luz estruturada, é a melhor solução adotada para o robô. A escolha bem criteriosa da técnica de medição garantiu o perfeito funcionamento do sistema como um todo.

O sistema de visão implementado demonstra ser eficiente. A geração de imagens de profundidade precisas é devido a uma série de algoritmos de tratamento e de correção sobre as imagens. Os algoritmos foram feitos de modo a serem rápidos, reduzindo operações matemáticas com números reais para minimizar o tempo de processamento.

O sistema computacional foi feito com a linguagem C#, de última geração, que é baseada em programação orientada a objetos (P.O.O.). Desta maneira, ela permitiu o seu desenvolvimento mais rápido, pois usou recursos deste paradgma de programação, como herança de classes, polimorfismo e encapsulamento.

O sistema automático de calibração para câmeras *CCD* é uma contribuição desta tese. Ele usa a transformação projetiva para a determinação dos parâmetros intrínsecos da câmera. A calibraçao é fácil e rápida, pois utiliza ambiente gráfico e com auxílio do *mouse*. Como recursos da calibracao, há a correção da distorção radial, que elimina o efeito barril causado pela lente da câmera e a interpolação bilinear, que define os valores dos *pixels* não mapeados durante o processo. Como resultado final, uma imagem em tons de cinza precisa e calibrada é obtida.

A calibração do plano de laser é automática, baseando-se no princípio da triangulação ativa, formada pela linha de laser e a câmera *CCD*. Há uma varredura sobre um objeto-padrão de

altura conhecida, com a perfeita sincronização entre obtenção de imagens e o movimento do laser. A modelagem matemática para a extração das cotas de altura dos objetos não requer cálculos complexos e tem baixo custo computacional.

Os parâmetros da calibração da câmera CCD e do plano de laser são guardados em arquivos e recuperados quando o sistema é reiniciado, mantendo o sistema sempre calibrado.

A geração de imagens de profundidade orto-retificadas é uma contribuição significativa da tese. A orto-retificação é comumente usada em imagens aéreas, para gerar orto-fotos. Contudo, ela foi adaptada para corrigir o efeito de perspectiva das imagens de profundidade. Esta correção é importante, especialmente porque as imagens do sistema são obtidas muito próximas dos obstáculos. Nesta caso, o efeito de perspectiva é mais evidente do que se as imagens fossem obtidas com obstáculos mais distantes. O algoritmo de orto-retificação corrige este efeito, obtendo-se assim imagens de profundidade precisas.

O sistema bem estruturado de referenciais é de fundamental importância para a integração total do sistema proposto com o sistema de referenciais do robô. A implementação do registro de imagens de profundidade sucessivas usa os dados odométricos do robô como estimativa inicial, cujas incertezas foram compensadas com o uso do algoritmo *ICP* de alinhamento de imagens, adaptado para imagens de profundidade.

A implementação de um algoritmo *ICP* customizado para imagens de profundidade, com uma escolha estratégica e criteriosa dos pontos correspondentes entre as imagens, garante alta convergência e redução do número de iterações, tornando-o mais rápido. Este algoritmo gera uma matriz de transformação que serve para corrigir a matriz de transformação da odometria do robô. Esta contribuição é muito importante, porque além de ajudar no alinhamento de imagens, o *ICP* também ajuda na redução do erro de odometria.

Após o processo de registro em que as imagens de profundidade são atualizadas em relação ao referencial do mundo, há milhares de pontos *3D*. A utilização destas imagens para

geração de descritores tornaria o processamento lento. Contudo, através da técnica da cubificação, somente pontos *3D* representativos são armazenados, reduzindo significativamente o esforço computacional, mas mantendo as características das imagens de profundidade.

A cubificação cria o ambiente propício para a extração de dados descritivos dos obstáculos. O processo de segmentação de obstáculos baseia-se em analisar na lista total de obstáculos elementares, aqueles que são vizinhos e eles são separados em outra lista, ordenados de forma crescente pelo identificador do obstáculo. A cada obstáculo é associada uma cor.

Uma característica interessante do sistema na geração dos descritores, é que eles são gerados em função da orientação do robô quando estão próximos dele. Além do mais, as imagens de profundidade podem obter mais detalhes da cena. Por exemplo, áreas de sombra de obstáculos, que são mapeados em uma segunda passagem do robô no local. A cubificacao é dinâmica, ou seja, a nova imagem de profundidade é cubificada eliminando os pontos redundantes da lista de obstáculos cubificados. Os novos descritores são todos atualizados com estes novos dados, e não há descritores redundantes para o mesmo obstáculo mapeado.

O mapeamento da superfície é formado pela lista de obstáculos cubificados. Este mapeamento é salvo em arquivos e recuperado quando necessário. É possível, também, juntar vários arquivos de mapeamento, e, assim, obter o mapeamento de superfícies maiores. Todos os obstáculos cubificados têm as coordenadas em relação ao referencial do mundo, o que facilita a integração de várias listas de obstáculos mapeados.

O conjunto de distâncias e pontos estratégicos do mapeamento é um rico conjunto de dados para usar em várias técnicas de controle de robôs quadrúpedes, porque mapeiam as distâncias das patas até os obstáculos no entorno do robô em 4 direções. O uso de círculos de procura minimiza o processamento, pois somente os obstáculos dentro ou tocando estes círculos são processados.

Para o uso do mapeamento da superfície e seus dados, é criada uma arquitetura de controle baseada em navegação reativa, que usa o conceito da *percepção e ação*. Esta técnica implementa uma árvore binária de decisão, para os casos de desvio ou transposição de um obstáculo.

As tomadas de decisão implementadas trouxeram bons resultados de navegabilidade para o robô Guará, mostrando-se eficiente para desviar ou transpor o obstáculo e trazendo-o de volta para a trajetória definida.

A implementação do controle baseado em navegação reativa cerca apenas alguns casos de situação robô/obstáculo, pois o principal objetivo é verificar a aplicabilidade do mapeamento da superfície na navegação do robô Guará.

No geral, o mapeamento da superfície é eficiente e preciso, tornando-se uma ferramenta muito útil para a navegação de robôs quadrúpedes.

## 7.2 Melhorias Futuras

Como melhorias futuras do sistema de visão a laser, a potência do laser poderia ser elevada de 3,5 mW para 5,0 mW para maior contraste do perfil do laser sobre os objetos mais escuros. O sistema atual aumenta o contraste, durante a conversão *RGB* para tons de cinza, enfatizando a componente Vermelha (*Red*).

A câmera *CCD*, na resolução máxima de 1280 x 1024, gera uma taxa de aproximadamente 15 quadros por segundo. Este valor baixo limita a varredura do laser sobre a superfície a uma velocidade mais baixa. Uma câmera com uma taxa de 60 quadros por segundo aumentaria significantemente a velocidade de aquisição do sistema.

A extração do perfil de laser é feita usando subtração de imagens da superfície com laser e somente da superfície. Este processamento poderia ser removido caso fosse instalado um filtro polarizado passa-faixa na câmera CCD, para o comprimento de onda do laser, que é

de 635 *nm*. Além do mais, um filtro polarizado reduz o problema de espalhamento do feixe de laser quando ele incide sobre objetos muito brilhantes.

Durante a navegação, o robô acumula erros de odometria. O algoritmo *ICP* ajuda a reduzi-los, mas, para distâncias maiores, este erro é significativo. A inclusão de marcadores de superfície (*land markers*) de cores ou formas específicas é uma boa alternativa para zerar o erro de odometria do robô Guará. Esta melhoria é de simples implementação porque a câmera *CCD* detectaria estes marcadores e a correção da odometria seria por transformações geométricas.

O robô Guará tem limites cinemáticos restritos, que para mudá-los necessitaria de uma reforma estrutural completa. A medição dos ângulos das juntas usa potenciômetros de precisão. A mudança destes sensores para *encoders* aumenta a precisão destas medidas, reduzindo os erros de odometria.

No que tange à arquitetura de controle baseada em navegação reativa, o algoritmo pode ser incrementado para tomadas de decisão sobre múltiplos obstáculos em múltiplas posições. Este trabalho, porém, é de grande complexidade, pois métodos não clássicos de controle devem ser usados.

# Bibliografia

[1] R.C. Gonzales and R. E. Woods, *Digital Image Processing*, 2nd edition, New Jersey: Prentice Hall, 1992.

[2] A. Bento Filho*, Modelagem, Controle de Andadura e Transposição de Obstáculos de um Robô Quadrúpede com Quatro Graus de Liberdade em Cada Perna*, Tese de Doutorado, Departamento de Engenharia Elétrica, Universidade Federal do Espírito Santo, Brasil, 2007.

[3] L. E. M. Lima, *Uma Arquitetura de Controle Hierárquico para um Robô Quadrúpede, com Comportamentos Reflexivos de Estabilidade, baseado em um Controlador Nebuloso com uso de um Acelerômetro*, Tese de Doutorado, Departamento de Engenharia Elétrica, Universidade Federal do Espírito Santo, Brasil, 2007.

[4] A. Bento Filho, P. F. S. Amaral, B. G. M Pinto, L. E. M. Lima, Uma Metodologia para a Localização Aproximada de um Robô Quadrúpede. *Anais do XV CBA*, Setembro 2004, Gramado, RS, Brasil.

[5] H. R. Everett, *Sensors for Mobile Robots, Theory and Application*, Massachusetts: A. K. Peters, Ltd, 1995.

[6] M.W. Spong and M. Vidyasagar, *Robot Dynamics and Control*, USA*:* John Wiley and Sons 1989.

[7] K. D. Harris and M. Recce, Experimental Modelling of Time-of-flight Sonar, *Elsevier Robotics and Autonomous Systems*, vol. 24, pp. 33-42, 1998.

[8] K. W. Jörg and M. Berg, Sophisticated Mobile Robot Sonar Sensing with Pseudo-random Codes, *Elsevier Robotics and Autonomous Systems*, vol. 25, pp. 241-251, 1998.

[9] G. Benet, M. Martínez, F. Blanes, P. Pérez and J. E. Simó, Differentiating Walls from Corners using the Amplitude of Ultrasonic Echoes, *Elsevier Robotics and Autonomous Systems*, vol. 50, pp. 15-25, 2005.

[10] P. L. Teoh, B. Shirinzadeh, C. W. Foong and G. Alici, The Measurement Uncertainties in the Laser Interferometry-based Sensing and Tracking Techique, *Elsevier Measurement*, vol. 32, pp. 135-150, 2002.

[11] Z Zhang, Determining the Epipolar Geometry and its Uncertainty: a Review, *International Journal of Computer Vision*, vol. 27, n. 2, pp. 161-198, 1998.

[12] X. Armangué and J. Salvi, Overall View Regarding Fundamental Matrix Estimation, *Elsevier Image and Vision Computing*, vol. 21, pp. 205-220, 2003.

[13] X. Armangué, H. Araújo and J. Salvi, A Review on Egomotion by Means of Differential Epipolar Geometry Applied to the Movement of a Mobile Robot, *Pergamon Pattern Recoginition*, vol. 36, pp. 2927-2944, 2003.

[14] C. Chen, Y. Hung, C. Chiang and J. Wu, Range Data Acquisition Using Color Structured Lighting and Stereo Vision, *Elsevier Image and Vision Computing*, vol. 15, pp. 445-456, 1997.

[15] J. Lobo, C. Queiroz and J. Dias, World Feature Detection and Mapping Using Stereo Vision and Inertial Sensors, *Elsevier Robotics and Autonomous Systems*, vol. 44, pp. 69-81, 2003.

[16] B. K. Horn and B. G. Schunck, Determining Optical Flow, *Artificial Intelligence*, vol. 17, pp. 185-203, 1981.

[17] J. L. Barron and S. S. Beauchemin, The Computation of Optical Flow, *ACM Computing Surveys*, vol. 27 (3), pp. 433-467, 1995.

[18] Y. Q. Shi and H. Sun, Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms and Standards, *CRC*, Boca Raton, 480p, 2000.

[19] F. Zhou and G. Zhang, Complete Calibration of a Structured Light Stripe Vision Sensor through Planar Target of Unknown Orientations, *Elsevier Image and Vision Computing*, vol. 23, pp. 59-67, 2005.

[20] Z. Wei, F. Zhou and G. Zhang, 3D Coordinates Measurement based on Structured Light Sensor, *Elsevier Sensors and Actuators*, vol. 120, pp.527-535, 2005.

[21] G. Zhang and Z. Wei, A Novel Calibration Approach to Structured Light 3D Vision Inspection, *Elsevier Optics & Laser Technology*, vol. 34, pp. 373-380, 2002.

[22] B. Zhang, Y. F. Li and Y. H. Wu, Self-recalibration of a Structured Light System via Plane-based Homography, *Elsevier Pattern Recognition*, vol. 40, pp. 1368-1377, 2007.

[23] G. Wang, Z. Hu, F. Wu and H. Tsui, Implementation and Experimental Study on Fast Object Modelling Based on Multiple Structured Stripes, *Elsevier Optics and Lasers in Engineering*, vol. 42, pp. 627-638, 2004.

[24] J. L. Posdamer and M. D. Altschuler, Surface Measurement by Space-coded Projected Beam Systems*, Computer Graphics and Image Processing*, vol. 18, pp. 1-17, 1982.

[25] J. Battle, E. Mouaddib and J. Salvi, Recent Progress in Coded Strutured Light as a Technique to Solve the Correspondence Problem: Survey, *Elsevier Pattern Recognition*, vol. 31, pp. 963-982, 1998.

[26] Y. Hsieh, Decoding Structured Light Patterns for Three-dimensional Imaging Systems, *Elsevier Pattern Recognition*, vol. 34, pp. 343-349, 2001.

[27] E. García and H. Lameda, A Low Power Laser Rangefinder for Autonomous Robot Application, *IECON*, vol. 1, 1996.

[28] E. García, H. Lameda and J. R. López, *A 3-D Vision System for Autonomous Robot Applications Based on a Low Power Semiconductor Laser Rangefinder*, Departamiento de Ingeniería Eléctrica, Electrónica y Automática, Universidad Carlos II, Madrid, 1996.

[29] C. Mertz, J. Kozar, J. R. Miller and C. Thorpe, Eye-safe Line Striper for Outside Use, *IEEE Intelligent Vehicle Symposium*, vol. 2, pp. 507-512, 2002.

[30] K. Hattori and Y. Sato, Handy Rangefinder for Active Robot Vision, *IEEE International Conference on Robotics and Automation*, pp. 1423-1428, 1995.

[31] K. Hattori and Y. Sato, Accurate Rangefinder with Laser Pattern Shifting, *IEEE Proceedings of ICPR*, pp. 849-853, 1996.

[32] C. F. Bergh, B. A. Kennedy, L. H. Matthies, A. E. Johnson, *A Compact, Low Power Two-axis Scanning Laser Rangefinder for Mobile Robots*, Jet Propulsion Laboratory, California Institute of Technology, USA, 2000.

[33] J. Forest and J. Salvi, A Review of Laser Scanning Three-dimensional Digitisers, *IEEE Proceedings of RSJ, International Conference on Intelligent Robots and Systems*, Switzerland, 2002.

[34] J. Haverinen and J. Röning, A 3-D Scanner Capturing Range and Color, *Scandinavian Symposium on Robotics*, Oulu, Finland, 1999.

[35] V. Sequeira, J. G. M. Gonçalves, M. I. Ribeiro, 3D Environment Modelling Using Laser Range Sensing, *Robotics and Autonomous Systems* 16, pp. 81-91, 1995.

[36] E. García and H. Lamela, Experimental results of a 3-D vision system for autonomous robot applications based on a low power semiconductor laser rangefinder. IEEE *Proceedings of the 24th Annual Conference of Industrial Electronics Society*, vol. 3 pp. 1338-1341, Germany, 1998.

[37] R. F. Vassallo, *Uma Estratégia de Navegação para Robôs Móveis em Ambientes Interiores*, Dissertação de Mestrado, Departamento de Engenharia Elétrica, Universidade Federal do Espírito Santo, Brasil, 1998.

[38] F. Shu, L. Toma, W. Neddermeyer and J.Zhang, Precise Online Camera Calibration in a Robot Navigation Vision System, *IEEE Proceedings of the International Conference on Mechatronics and Automation*, pp 1277-1282, 2005.

[39] F. Zheng and B. Kong, Calibration of Linear Structured Light System by Planar Checkboard, *IEEE International Conference on Information Acquisition*, pp. 344-346, 2004.

[40] G. Seetharaman, H. Bao and G. Shivaram, A Fast and Simple Method to Calibrate Scale Factor using Telephoto Lens, *IEEE Proceedings of 10th International Symposium on Intelligent Control*, pp 326-331, 1995.

[41] P. Heckbert, Projective Mappings for Image Warping, *Image-Based Modeling and Rendering,* graduate course n. 15-869, Carnegie Mellon University, USA, 1999.

[42] H. G. Nguyen and M. R. Blackburn, A Simple Method for Range Finding via Laser Triangulation, *NCCOSC Technical Document*, TD-2734, Naval Command Control an Ocean Surveillance Center, San Diego CA, USA 1995.

[43] Z. Qin, W. Li, M. Li, Z. Chen and G. Zhou, A Methodology for True Orthorectification of Large-Scale Urban Aerial Images and Automatic Detection of Building Occlusions Using Digital Surface Maps, *IEEE Proceedings of the International Geoscience and Remote Sensing Symposium,* pp 729-731, 2003.

[44] Orlandi, J. G. N. and P. F. S. Amaral, Generation of Orthorectified Range Images for Robots Using Monocular Vision and Laser Stripes, *Latin American Applied Research International Journal*, vol. 38, pp 27-33, 2008.

[45] Blais G. and M. D. Levine, Registering Multiview Range Data to Create 3D Computer Objects, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp 820-824, 1995.

[46] Roth G., Registering two overlapping range images, *Proceedings of the Second International Conference on 3-D Digital Imaging and Modeling*, pp 191-200, Canada, 1999.

[47] S. Rusinkiewicz and M. Levoy, Efficient Variants of the ICP Algorithm, *IEEE Proceedings of the Third International Conference on 3-D Digital Imaging and Modelling*, pp 145-152, 2001.

[48] P. J. Besh and N. D. McKay, *A* Method for Registration of 3-D Shapes, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp 239-256, 1992.

[49] M. Y. Ibrahim and A. Fernandes, Study on Mobile Robot Navigation Techniques, *IEEE International Conference on Industrial Technnology*, vol. 1, pp 230-236, 2004.

[50] J. Minguez and L. Montano, Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios, *IEEE Transactions on Robotics and Automation*, vol. 20, pp 45-59, 2004.

[51] H. Igarashi, S. Shirasaka, S. Suzuki and M. Kakikura, Teleoperation System for Multiple Robots with HAM: Free Gait Following Motion of Quadruped Robots, *IEEE Internation Workshop on Robots and Human Interactive Communication*, pp 484-489, 2005.

[52] K. Izumi, K. Watanabe, M. Shindo and R. Sato, Acquisition of Obstacle Avoidance Behaviors for a Quadruped Robot Using Visual and Ultrasonic Sensors, *IEEE 9th*

*International Conference on Control, Automation, Robotics and Vision, ICARCV*, pp 1-6, 2006.

[53] T. Mitchell, *Machine Learning*, McGraw Hill, 1997.

[54] J. Minguez, J. Osuna and L. Montano, A Divide and Conquer Strategy based on Situations to Achieve Reactive Collision Avoidance in Troublesome Scenarios, *IEEE International Conference on Robotics and Automation, ICRA 2004*, vol. 4, pp 3855- 3862, 2004

# Livros Grátis

( http://www.livrosgratis.com.br )

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo