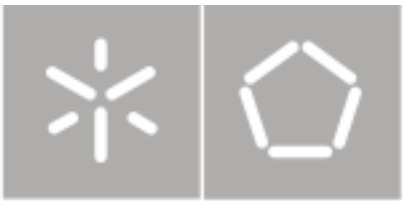
Universidade do Minho
Escola de Engenharia

João Miguel Clemente de Sena Esteves

Metodologia de Autolocalização Absoluta em Ambientes Quase-Estruturados

Julho de 2005



Universidade do Minho
Escola de Engenharia

João Miguel Clemente de Sena Esteves

Metodologia de Autolocalização Absoluta em Ambientes Quase-Estruturados

Tese submetida à Universidade do Minho para a obtenção do grau de Doutor em Engenharia Electrónica Industrial

Trabalho efectuado sob a orientação de

Professor Doutor Carlos Alberto C. Monteiro e Couto
Departamento de Electrónica Industrial da Escola de Engenharia da Universidade do Minho

Professor Doutor Adriano da Silva Carvalho
Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Julho de 2005

DECLARAÇÃO

Nome: JOÃO MIGUEL CLEMENTE DE SENA ESTEVES

Título da Tese de Doutoramento:

Metodologia de Autolocalização Absoluta em Ambientes Quase-Estruturados

Ano de conclusão: 2005

Universidade do Minho, 28/07/2005

Assinatura

Agradecimentos

Agradeço ao Professor Doutor Carlos Alberto Caridade Monteiro e Couto e ao Professor Doutor Adriano da Silva Carvalho a orientação e o apoio prestados no decorrer deste trabalho. E a todos os colegas e amigos que tanto contribuíram com a sua amizade e incentivo. Em especial, ao Doutor Manuel João Sepúlveda Mesquita de Freitas, meu colega de gabinete, pela sua proverbial paciência. À Doutora Filomena Maria da Rocha Menezes Oliveira Soares, do gabinete ao lado, pelo mesmo motivo e também pela leitura da tese. À Doutora Estela Erlhagen e ao Doutor Luís Ribeiro, que se ofereceram para o fazer. Ao Engenheiro José Cabral e à Doutora Graça Minas pela ajuda nas capas da tese. Ao Engenheiro José Joaquim Carvalho pelas cópias das teses. Ao Doutor Miguel Pupo Correia, à Engenheira Susana Carneiro, à Doutora Margarida Ferreira, à Doutora Ana Maria Faustino, ao Engenheiro Carlos Machado, ao Doutor Pedro Oliveira, à Doutora Celina Pinto Leão, ao Doutor Adriano Tavares, ao Doutor Carlos Silva, ao Doutor Carlos Lima, ao Doutor Manuel João Ferreira, à Doutora Arminda Gonçalves e à Doutora Daniela Castro pelos esclarecimentos nas respectivas áreas de competência. À Doutora Ana Isabel Cabrita pelo *Manual de Navegação*. À Sofia Oliveira Martins pelos dados sobre Pothenot. À Engenheira Ana Maria Augusto, à Engenheira Isabel Cardoso e à Doutora Maria Inês Carvalho pela disponibilidade. Ao Engenheiro Renato Morgado, pela colaboração na disciplina de Electrotecnia e ao Doutor José Augusto Afonso, pelo apoio na disciplina de Práticas de Electrónica.

Por todo o apoio com que me brindaram de forma habitual, agradeço aos meus amigos Isaac Fernández-Jardon, José Gaspar Fiuza Branco, Amadeu e Maria Felicidade Mesquita Guimarães, Artur e Ana Paula Mesquita Guimarães, António e Anabela Bouça, Pedro e Maria Manuel Menezes da Silva, Rui e Sofia Ribeiro, João Paulo e Graça Sousa, Luís e Amélia Martelo, Diana Mendes, Casimiro Sarroeira, Fernando Vasco Cardoso, António Leitão, Sílvia Reis, Luís Guimarães, Catarina Martins da Rocha, André Soares, José Paulo Duque, Pedro Pimentel, Helena Tavares, Maria José Sousa e, em especial, Sofia Oliveira Martins, pelos muitos incentivos via SMS.

Mais do que a todos, estou muito agradecido aos meus pais – a quem devo noventa por cento do que sou – e também à minha avó, irmãos, cunhado e sobrinhos.

A meus pais

Resumo

A localização absoluta com balizas é a solução mais adequada ao desenvolvimento de um sistema fiável, exacto e de custo relativamente baixo para a localização contínua em tempo real de robôs móveis que navegam, com velocidades de alguns metros por segundo, em ambientes (exteriores ou interiores) quase-estruturados e não muito obstruídos.

Entre os métodos que recorrem a balizas, a autolocalização absoluta por triangulação possui vantagens importantes quando se pretende localizar simultaneamente vários robôs que navegam a duas dimensões, desde que a navegação se faça sobre superfícies regulares e sem desníveis pronunciados.

O Algoritmo de Triangulação Geométrica está sujeito às limitações inerentes a qualquer algoritmo de autolocalização por triangulação com três balizas mas, além disso, requer uma ordenação especial de balizas e só funciona de forma consistente quando o robô se encontra dentro do triângulo formado por três balizas não colineares. Por estas razões tem sido considerado de menor interesse por vários autores. No entanto, quando comparado com os seus congéneres, o algoritmo possui vantagens que justificam a tentativa de eliminar as suas limitações específicas. Esse trabalho foi realizado, dando origem ao Algoritmo Generalizado de Triangulação Geométrica.

A generalização do Algoritmo de Triangulação Geométrica teve por base um quadro de análise da autolocalização absoluta por triangulação com três balizas. Este inclui um novo método que permite, em tempo real, caracterizar incertezas de posição e de orientação e detectar situações nas quais a localização não é possível.

Apresentam-se os resultados obtidos em quatro conjuntos de testes realizados – mediante a simulação por computador – com o fim de analisar e validar o Algoritmo Generalizado de Triangulação Geométrica e o novo método de caracterização das incertezas de posição e de orientação.

Palavras-Chave: Autolocalização absoluta, localização com balizas, medição de ângulos, triangulação, problema de *Pothenot*, análise de erros, robótica móvel, localização de robôs.

Abstract

Absolute localization with beacons is the most adequate solution regarding the development of a reliable, accurate and relatively low cost real-time continuous localization system for mobile robots navigating, at a speed of few meters per second, in quasi-structured and not very obstructed interior or exterior environments.

Among the methods that use beacons, absolute self-localization by means of triangulation has important advantages when it is necessary to simultaneously locate several robots navigating on a plane, as long as navigation occurs over even surfaces without important level changes.

The Geometric Triangulation Algorithm has the restrictions that are inherent to any self-localization three-beacon triangulation algorithm but, in addition, it requires a particular beacon ordering and only works consistently when the robot lies inside the triangle formed by three non-collinear beacons. For these reasons, it has been disregarded by several authors. However, when compared to other triangulation algorithms, this one has some advantages that justify the attempt to eliminate its specific restrictions. This has been done, originating the Generalized Geometric Triangulation Algorithm.

The generalization of the Geometric Triangulation Algorithm was based on an analysis framework of absolute self-localization by means of triangulation with three beacons. This analysis framework includes a new real-time method of determining the uncertainties associated with the computed position and orientation, and of detecting situations in which localization is not possible.

This work also presents the results obtained in four tests made – using computer simulation – in order to analyze and validate the Generalized Geometric Triangulation Algorithm and the new method of determining position and orientation uncertainties.

Keywords: Absolute self-localization, localization with beacons, angle measurements, triangulation, *Pothenot's* problem, error analysis, mobile robotics, robot localization.

Índice

1. Introdução

Em 1617, no seu trabalho *Eratosthenes Batavus*, o holandês Willebrord Snellius (1581-1626) enunciou e resolveu um problema que viria a ser de grande utilidade na Topografia e na Navegação: *Determinar a posição de um ponto P desconhecido e acessível a partir das orientações, medidas a partir de P, de três pontos A, B e C inacessíveis e conhecidos.* O problema foi novamente resolvido pelo francês Pothenot (falecido em 1732), apareceu num artigo submetido à Academia Francesa em 1692, e só depois se tornou do conhecimento comum, sob o nome de *Problema de Pothenot*[1] (Dorrie, 1965). Desde então, tem sido muito utilizado na realização de levantamentos topográficos e também na navegação costeira de embarcações.

As medidas dos ângulos formados pelos segmentos de recta que unem uma embarcação a três *conhecenças*[2] (Figura 1.1) permitem traçar, numa carta náutica, dois arcos de circunferência que se intersectam no ponto correspondente à posição da embarcação (Figura 1.2). Este pode determinar-se mais rapidamente recorrendo a um compasso de três pontas (Figura 1.3). Uma vez determinada a posição da embarcação, a sua orientação calcula-se facilmente a partir da medida do ângulo formado por um eixo de referência fixo na embarcação com o segmento de recta que une a embarcação a uma das conhecenças (Figura 1.1). Para evitar significativos erros de medição de ângulos, as conhecenças não devem estar em planos muito diferentes (Marques, 2001). Como é do conhecimento dos navegantes, não é possível determinar univocamente a posição da embarcação quando esta se encontra sobre a circunferência definida por três conhecenças não colineares[3]. Por este motivo, no *Manual de Navegação* do Instituto Hidrográfico (MN, 1989) só se garante que a posição fica bem determinada se a embarcação se encontrar dentro do triângulo formado pelas três conhecenças ou se a conhecença central ficar aquém do segmento de recta que une as outras duas. Nessas circunstâncias, a embarcação não pode estar sobre a referida circunferência (Figura 1.4).

[1] Este problema também é conhecido por *Trisecção Inversa, Problema dos Três Pontos*, ou *Problema de Vértice de Pirâmide*, na Topografia, ou ainda *Problema da Carta*, na Navegação (Espartel, 1980; García-Tejero, 1981).

[2] Uma *conhecença*, ou *ponto conspícuo*, é um objecto simultaneamente bem visível do mar e assinalado nas cartas náuticas (Marques, 2001).

[3] A circunferência degenera numa recta, se as conhecenças forem colineares.

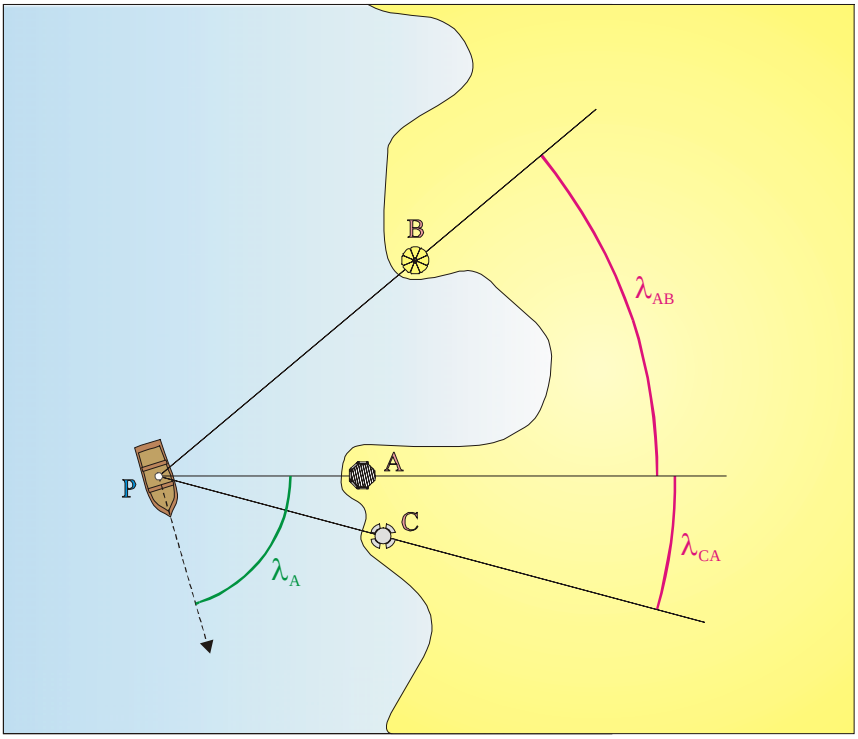


Figura 1.1: A posição da embarcação pode ser determinada a partir das medidas dos ângulos λ_{AB} e λ_{CA}. Depois, a sua orientação pode determinar-se a partir da medida de λ_A.

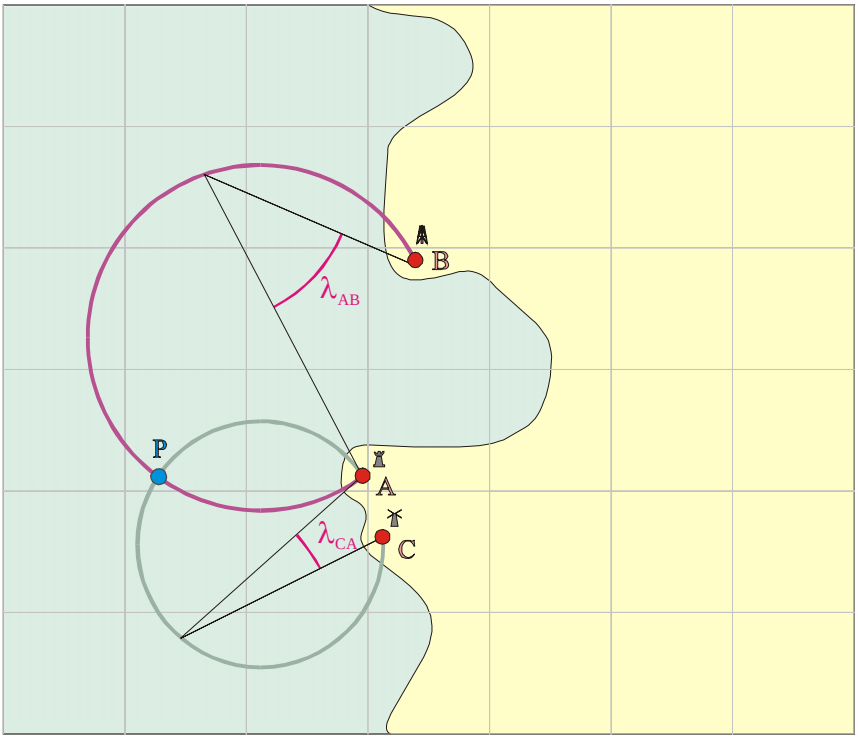


Figura 1.2: As medidas dos ângulos λ_{AB} e λ_{CA} permitem traçar, numa carta náutica, dois arcos de circunferência que se intersectam no ponto correspondente à posição da embarcação.

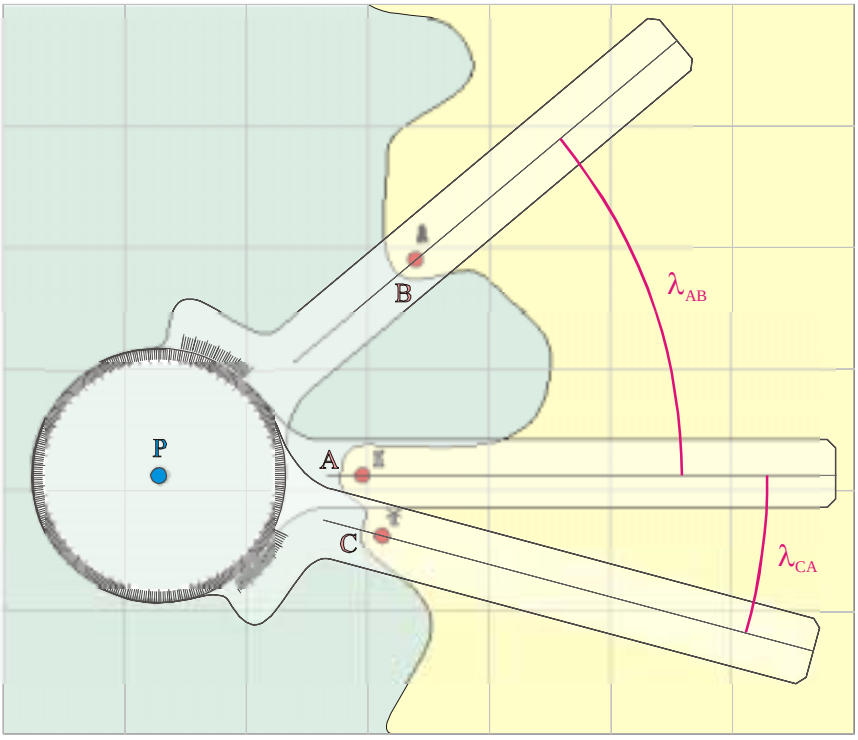


Figura 1.3: Utilização de um compasso de três pontas para determinar, numa carta náutica, o ponto correspondente à posição de uma embarcação.

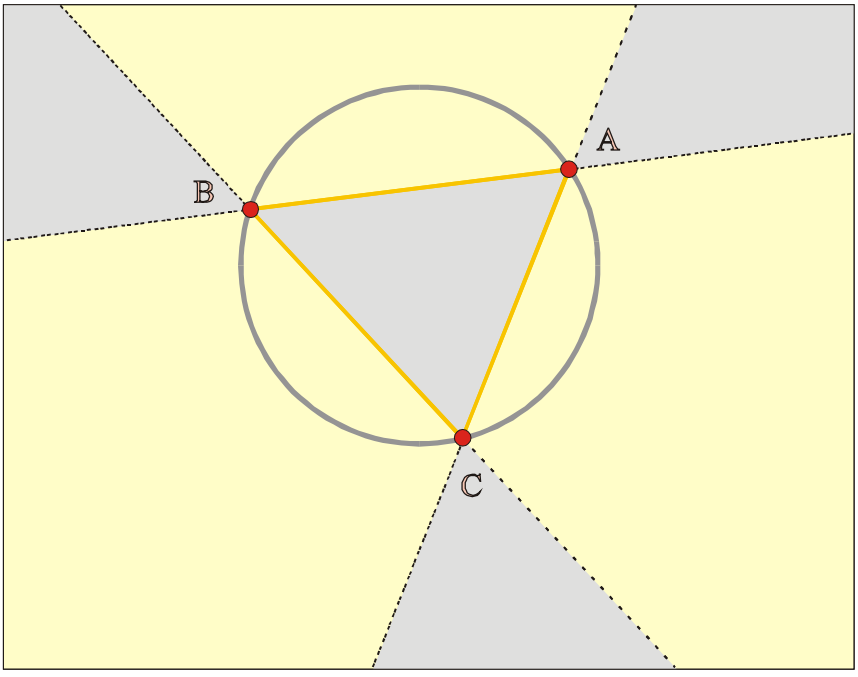


Figura 1.4: Uma embarcação nunca se situa sobre a circunferência definida por três conhecenças não colineares desde que permaneça no interior de alguma das regiões sombreadas a cinzento: o interior do triângulo formado pelas três conhecenças e as regiões tais que a conhecença central se encontra aquém do segmento de recta que une as outras duas.

1.1 Motivação e Objectivos

O aparecimento do computador tornou viável a resolução por via analítica do Problema de Pothenot para efeitos de *autolocalização*[4] em tempo real. No âmbito da robótica[5] móvel chama-se *autolocalização absoluta*[6] *por triangulação*[7] à que se baseia na resolução do Problema de Pothenot, e há muitos algoritmos disponíveis para o efeito (Dorrie, 1965; Espartel, 1980; Davis *et al.*, 1981; García-Tejero, 1981; Mcgillem e Rappaport, 1989; Cohen e Koss, 1992; Everett, 1995; Fuentes *et al.*, 1995; Stella *et al.*, 1995; Borenstein *et al.*, 1996; Betke e Gurvits, 1997; Ji *et al.*, 2003; Sena Esteves *et al.*, 2003). O interesse neste tipo de autolocalização é suscitado pelo bom desempenho (fiabilidade, exactidão) e custo relativamente baixo que actualmente caracterizam diversos sistemas de localização que utilizam balizas.

Um estudo efectuado sobre os métodos de localização habitualmente utilizados na robótica móvel, em particular dos que recorrem a balizas (os resultados são apresentados no Capítulo 2 e no Capítulo 3), revela que

- os métodos que recorrem a balizas são adequados a muitas aplicações que requerem a localização contínua em tempo real de robôs móveis que navegam a duas dimensões, com velocidades de alguns metros por segundo, em ambientes (exteriores ou interiores) quase-estruturados não muito obstruídos.

  Esta descrição aplica-se a vários ambientes – encontrados, por exemplo, em fábricas, portos, hospitais ou quintas (De Cecco, 2002) – nos quais circulam *AGVs (Automatic Guided Vehicles)* que se movem a velocidades até 1m/s (Castleberry, 1991; SIEMENS, 2002).

- a autolocalização absoluta por triangulação apresenta, relativamente aos outros métodos de localização com balizas, importantes vantagens quando se pretende localizar simultaneamente vários robôs que navegam a duas dimensões, desde que não ocorram inclinações significativas desses robôs.

---

[4] A *localização* de um veículo consiste na determinação da sua posição e da sua orientação relativamente a um dado referencial. Normalmente considera-se que a posição de um veículo é a posição de um dos seus pontos. Dá-se o nome de *autolocalização* à localização feita a partir do próprio veículo.

[5] *"Robô: qualquer máquina que, funcionando automaticamente, substitui o esforço humano, apesar de poder não ser como os humanos na aparência ou não funcionar de uma maneira semelhante à dos humanos"*. (TNEB, 1990).

[6] Que não recorre a suposições sobre movimentos anteriores do robô.

[7] Que recorre exclusivamente à medição de ângulos.

- a tecnologia actual, nomeadamente a dos sistemas ópticos com balizas activas, permite obter uma exactidão de medição de posição da ordem dos milímetros e uma exactidão de medição de orientação da ordem das centésimas de grau.

  A exactidão de medição de posição requerida para um AGV num posto de recolha e entrega é da ordem dos milímetros (Castleberry, 1991). E quando se pretende alinhar um AGV com uma estrutura – por exemplo uma plataforma – é particularmente importante considerar a exactidão na medição de orientação. Basta um erro de orientação de 2º para que um dos vértices de um AGV rectangular com 3m de comprimento se encontre 10cm mais afastado da estrutura do que o outro vértice (Figura 1.5). Para limitar esta diferença a cerca de 1mm, o erro de orientação não pode – neste exemplo – ultrapassar 0,02º. Em geral, considerando o comprimento de um AGV típico, para limitar os erros de posição a alguns milímetros em todos os seus pontos, é necessário limitar os erros de orientação às centésimas de grau. Como se ilustra na Figura 1.5, os erros de orientação são independentes de eventuais erros de posição.

O Algoritmo de Triangulação Geométrica (Cohen e Koss, 1992), que resolve o Problema de Pothenot e também permite que um veículo determine a sua orientação, tem sido considerado de menor interesse por só funcionar coerentemente dentro do triângulo definido por três balizas. Este algoritmo tem sido preterido em favor de outros que, por sua vez, também possuem limitações. As limitações específicas destes últimos são inerentes aos próprios métodos, e por isso mesmo inultrapassáveis. No entanto, mostrar-se-á que o Algoritmo de Triangulação Geométrica pode ser generalizado por forma a conservar apenas as limitações que são comuns a qualquer algoritmo de autolocalização por triangulação e possui características interessantes que justificam esse esforço. O primeiro objectivo deste trabalho é o de eliminar as suas limitações específicas, dando origem ao Algoritmo Generalizado de Triangulação Geométrica.

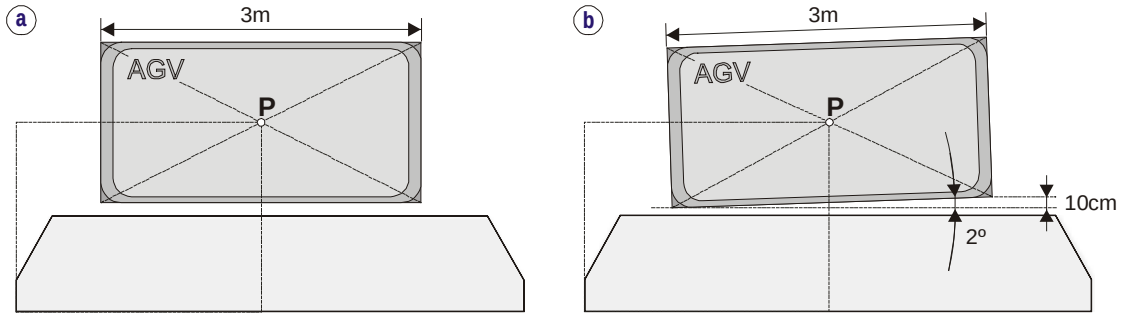


Figura 1.5: Alinhamento de um AGV rectangular de 3m de comprimento com uma estrutura (considera-se que a posição do AGV é a posição do ponto P). a) O AGV está alinhado com a estrutura, pelo que não há erro de orientação. b) Basta um erro de orientação de 2º para que um dos vértices do AGV se encontre 10cm mais afastado da estrutura do que o outro vértice; apesar disso, a posição do ponto P é a mesma que ocorre em a), com o AGV devidamente alinhado com a estrutura.

Em geral, a posição calculada não coincide com a posição verdadeira do robô e a orientação calculada também é diferente da sua orientação verdadeira. Existem um erro de posição e também um erro de orientação que se devem, sobretudo, aos erros de medição dos ângulos e à posição do robô relativamente às balizas.

Na prática, a posição e a orientação calculadas são inúteis para efeitos de navegação quando não se fazem acompanhar das respectivas incertezas. E é também necessário que sejam detectadas as situações nas quais a localização não é possível, de forma a que o veículo que se está a tentar autolocalizar possa realizar uma acção adequada a essa circunstância. Assim, o segundo objectivo proposto é desenvolver um método, aplicável ao Algoritmo Generalizado de Triangulação Geométrica capaz de, em tempo real:

1. caracterizar as incertezas associadas à posição e à orientação calculadas;

2. detectar situações nas quais a localização não é possível.

1.2 Organização da Tese

Depois deste capítulo introdutório far-se-á, no Capítulo 2, uma análise de métodos utilizados na localização de robôs móveis. Mostrar-se-á que, entre todos esses métodos, os que recorrem a balizas são os mais adequados ao desenvolvimento de um sistema fiável, exacto e de custo relativamente baixo para a localização contínua em tempo real de robôs móveis que naveguem a duas dimensões, com velocidades de alguns metros por segundo, em ambientes (exteriores ou interiores) quase-estruturados e não muito obstruídos.

No Capítulo 3 estudar-se-ão diversos modos de localizar um robô recorrendo a balizas. Mostrar-se-á que, de acordo com os aspectos avaliados, a autolocalização absoluta por triangulação, é a melhor solução para a localização simultânea de vários robôs que navegam a duas dimensões, desde que a navegação se faça sobre superfícies regulares e sem desníveis pronunciados.

No Capítulo 4 proceder-se-á à análise comparativa de diversos algoritmos de triangulação que têm sido usados no âmbito da robótica móvel para efectuar a autolocalização absoluta a duas dimensões.

No Capítulo 5 irá propor-se um quadro de análise da autolocalização absoluta por triangulação com três balizas, que constitui a base da generalização do Algoritmo de Triangulação Geométrica, mas é aplicável a outros algoritmos.

O primeiro passo consistirá numa cuidadosa definição de ângulos a utilizar em algoritmos de triangulação. Em concreto, definir-se-ão os ângulos necessários para:

- caracterizar a configuração de balizas;
- determinar sem ambiguidade a posição e a orientação do robô.

De acordo com estas definições de ângulos, será sugerida uma nova especificação do problema da autolocalização absoluta, a duas dimensões, por triangulação.

Apresentar-se-á um novo método que permite, em tempo real, caracterizar incertezas de posição e de orientação na autolocalização absoluta por triangulação com três balizas e detectar situações nas quais a localização não é possível. Pode ser usado para

- caracterizar a incerteza de posição e também a incerteza de orientação, se se partir do princípio que os erros de medição têm limites finitos conhecidos;
- caracterizar a incerteza de posição devida aos erros aleatórios de medição, se se considerar que esses erros possuem distribuições de probabilidade gaussianas.

No Capítulo 6 apresentar-se-á o Algoritmo Generalizado de Triangulação Geométrica que não requer ordenação de balizas e apenas está sujeito às restrições que são comuns a todos os algoritmos de autolocalização por triangulação.

Serão ainda apresentados os resultados obtidos em quatro conjuntos de testes, realizados – mediante a *simulação por computador*[8] – com as seguintes finalidades:

- Validar o Algoritmo Generalizado de Triangulação Geométrica;

[8] *"Computer simulation is the discipline of designing a model of an actual or theoretical physical system, executing the model on a digital computer, and analyzing the execution output"* (Fishwick, 1994).

- Verificar de que modo é que os erros de medição dos ângulos afectam os erros de posição e de orientação, quando o robô se encontra próximo ou afastado das balizas;

- Validar o novo método de caracterização das incertezas de posição e de orientação;

- Determinar, para várias posições do robô relativamente às balizas, o tempo necessário para calcular a sua posição, a sua orientação e as incertezas que lhes estão associadas, quando se recorre ao Algoritmo Generalizado de Triangulação Geométrica e ao novo método de caracterização das incertezas de posição e de orientação.

Por fim, no Capítulo 7 apresentar-se-ão as conclusões gerais do trabalho realizado e algumas perspectivas de trabalho futuro.

1.3 Contributos Científicos

Os principais contributos científicos desta tese são os seguintes:

- Um quadro de análise da autolocalização absoluta por triangulação com três balizas, que inclui:

  a) uma cuidadosa definição de ângulos a utilizar em algoritmos de triangulação;

  b) uma nova especificação do problema da autolocalização absoluta, a duas dimensões, por triangulação;

  c) um novo método que permite caracterizar, em tempo real, as incertezas de posição e de orientação na autolocalização absoluta por triangulação com três balizas. Pode ser usado quer se parta do princípio que os erros de medição têm limites conhecidos quer se considere que esses erros possuem distribuições de probabilidade gaussianas (no segundo caso, o método permite caracterizar apenas a incerteza de posição).

- A generalização do Algoritmo de Triangulação Geométrica, feita com base no quadro de análise proposto.

Além disso, são sugeridos:

- dois novos métodos que permitem a autolocalização recorrendo apenas a duas balizas;

- um algoritmo de triangulação simples, mas que consiste na resolução de um sistema de três equações não lineares;

- uma especificação do Algoritmo de Triangulação Baseado na Pesquisa Iterativa;

- uma especificação do Algoritmo de Triangulação Baseado no Método de Newton-Raphson.

2. Métodos de Medição da Posição e da Orientação de Robôs Móveis

A *navegação* pode definir-se como o *"conjunto de operações destinadas a situar e dirigir um veículo entre o ponto de partida e o de chegada"* (Bettencourt, 1972). Christou *et al.* (1992) identificam a navegação exclusivamente com a medição da posição do veículo e utilizam o conceito de *guia* para referir a determinação da informação direccional necessária para conduzir o veículo sem necessariamente calcular a sua posição. Em NDC (1998) define-se a navegação como o processo de determinar a posição e a orientação de um veículo, e usa-se o termo *regulação* para designar o processo de corrigir estes dois parâmetros. A navegação entendida desta forma confunde-se com o conceito de *localização* que será utilizado neste trabalho.

Aceitar-se-á aqui a definição de navegação proposta por Bettencourt (1972). Esta está de acordo com as dos autores que, no conceito de navegação, incluem não só a medição de posição mas também o planeamento do percurso a descrever e a condução do veículo de modo a evitar os obstáculos com que depare (McKerrow, 1991; Leonard e Durrant-White, 1992; Turennout e Honderd, 1992; Kortenkamp *et al.*, 1998).

Leonard e Durrant-White (1992) resumem o *problema da navegação* de um robô móvel (o veículo) a três questões colocadas do ponto de vista do robô:

- Onde estou?
- Para onde vou?
- Como fazer para lá chegar?

A primeira destas questões corresponde à *localização*, que definem como sendo o *"processo de determinar a posição de um robô num referencial global utilizando a informação proveniente de sensores externos"*. Os mesmos autores referem Drumheller, para quem o problema da localização envolve não só a medição de posição mas também a medição de orientação. Neste trabalho adoptou-se esta definição mais abrangente. O

mesmo fazem, por exemplo, McKerrow (1991)[1], Borenstein *et al.* (1996)[2], Cauchois *et al.* (2002), Gutmann (2002), Venet *et al.* (2002) e Briechle e Hanebeck (2004).

A *autolocalização* consiste em ser o próprio veículo a determinar a sua posição e a sua orientação. Na *localização remota* é um sistema exterior ao veículo que o localiza. A autolocalização é mais simples de implementar do que a localização remota quando é necessário localizar simultaneamente vários veículos.

Neste capítulo serão brevemente analisados os métodos de medição de posição e de orientação referidos na Figura 2.1. Todos se podem utilizar na autolocalização de robôs móveis. A triangulação e a trilateração com balizas e o reconhecimento de marcos naturais também se usam na sua localização remota. O objectivo da análise é encontrar o método mais adequado ao desenvolvimento de um sistema fiável e de custo relativamente baixo para a localização contínua, simultânea e em tempo real de vários robôs móveis que navegam a duas dimensões, com velocidades de alguns metros por segundo, em ambientes (exteriores ou interiores) quase-estruturados e não muito obstruídos. Pretende-se obter uma exactidão de medição de posição da ordem dos milímetros e uma exactidão de medição de orientação da ordem das centésimas de grau.

No ponto 2.1 analisam-se métodos usados na *localização absoluta*, ou seja, na determinação da posição e da orientação de um veículo de um modo independente de suposições sobre movimentos anteriores.

No ponto 2.2 analisam-se métodos usados na *localização relativa*, ou seja, no cálculo da posição e da orientação actuais de um veículo mediante a actualização de uma posição e orientação medidas anteriormente.

[1] Para definir a localização, McKerrow (1991) recorre a um vector com seis graus de liberdade que descreve a posição e a orientação de um objecto no espaço.

[2] Borenstein *et al.* (1996) também usam o termo *posicionamento* em sentido lato, para designar a tarefa de encontrar a posição e a orientação de robôs móveis.

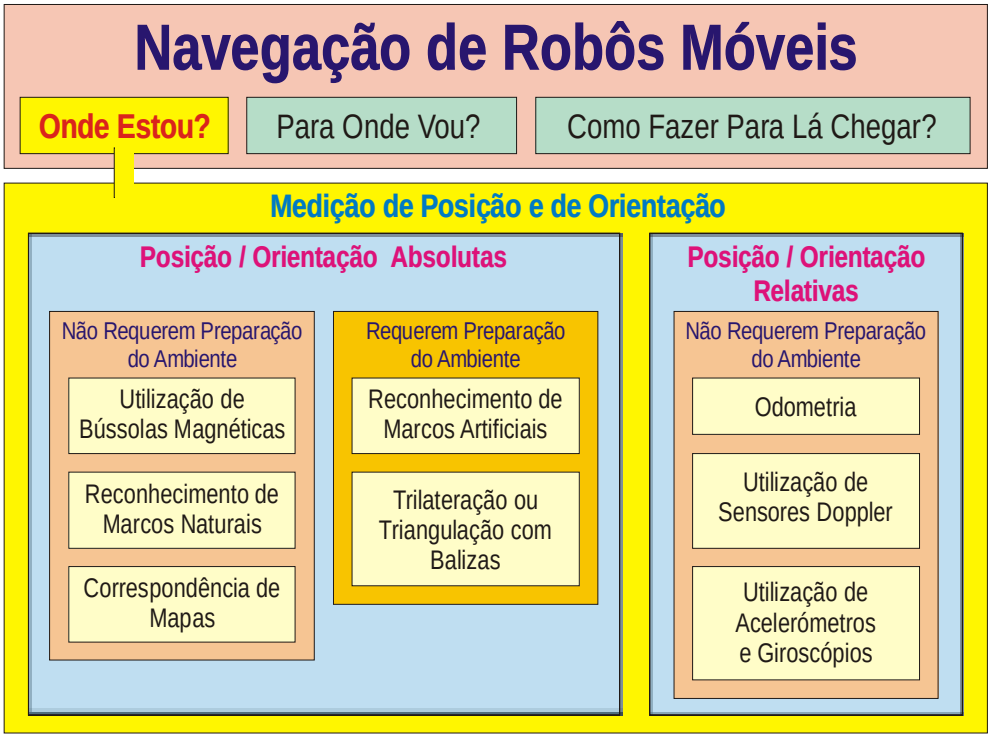


Figura 2.1: Métodos de medição da posição e da orientação de robôs móveis.

2.1 Medição de Posição e Orientação Absolutas

Segundo Drumheller, citado por Leonard e Durrant-White (1992), a *localização absoluta* refere-se à *"possibilidade de um robô móvel determinar a sua posição e orientação [...] de um modo independente de suposições sobre movimentos anteriores"*. A posição e a orientação assim determinadas designam-se *posição absoluta* e *orientação absoluta* (Aider *et al.*, 2002; De Cecco, 2002; Venet *et al.*, 2002; Hernández *et al.*, 2003).

2.1.1 Métodos que Não Requerem Preparação do Ambiente

Alguns métodos de localização absoluta requerem que o ambiente no qual um veículo se move seja previamente preparado para a navegação, como se verá mais adiante. A principal vantagem dos métodos que seguidamente se descrevem, habitualmente utilizados na autolocalização absoluta de robôs móveis, é a de dispensarem essa preparação. Para se localizar, o veículo capta características do próprio ambiente com o auxílio de sensores apropriados.

2.1.1.1 Utilização de Bússolas Magnéticas

As bússolas magnéticas são há muito utilizadas pelos navegadores para indicar a direcção do norte magnético, situado próximo do norte geográfico da Terra.

É preciso ter em conta a existência de campos magnéticos locais que podem ser provocados por (McKerrow, 1991; Borenstein *et al.*, 1996; Borenstein *et al.*, 1997; Jones *et al.*, 1999):

- linhas de transporte de energia;
- grandes massas de ferro;
- vigas de aço em edifícios;
- componentes metálicos do próprio veículo.

Estes campos magnéticos locais sobrepõem-se ao campo magnético terrestre e afectam as leituras de qualquer bússola magnética. A direcção do norte magnético só é indicada quando não existem campos magnéticos locais.

Tendo por base vários fenómenos físicos relacionados com o campo magnético da Terra, existem diversos tipos de bússolas, analisados detalhadamente por Everett (1995) e Borenstein *et al.* (1996):

- Bússolas magnéticas mecânicas;
- Bússolas *fluxgate*;
- Bússolas de efeito Hall;
- Bússolas magnetoindutivas;
- Bússolas magnetoresistivas;
- Bússolas magnetoelásticas.

As bússolas *fluxgate* são as mais indicadas para aplicações com robôs móveis (Borenstein *et al.*, 1996; Borenstein *et al.*, 1997). Quando mantidas na horizontal, estas

bússolas medem a componente horizontal do campo magnético terrestre, com as seguintes vantagens:

- Baixo consumo;
- Sem partes móveis;
- Toleram choques e vibrações;
- Arranque rápido;
- Custo relativamente baixo.

Borenstein *et al.* (1997) referem, para a *KVH Fluxgate Compass*, as características apresentadas na Tabela 2.1.

Tabela 2.1: Características da *KVH Fluxgate Compass*.

Resolução *(resolution)*	±0,5º
Exactidão *(accuracy)*	±0,5º
Repetibilidade *(repeatability)*	±0,2º

Quando funcionam em exteriores, as bússolas magnéticas são muito fiáveis e, depois de calibradas para o norte magnético local, também são exactas (Jones *et al.*, 1999). No entanto, devido à existência dos campos magnéticos locais, é difícil utilizar directamente sensores geomagnéticos nas aplicações em interiores, nomeadamente na navegação. Jones *et al.* (1999) consideram viável a utilização de bússolas magnéticas nessas circunstâncias se se puderem tolerar erros da ordem de ±45º.

2.1.1.2 Reconhecimento de Marcos Naturais

Os *marcos (landmarks)* são características nítidas do ambiente que podem ser reconhecidas por sensores adequados e utilizadas para efeitos de localização (Borenstein *et al.*, 1996; Borenstein *et al.*, 1997). Cada marco é cuidadosamente escolhido para ser facilmente identificável e tem geralmente uma posição fixa e conhecida, relativamente à qual o robô se pode localizar. As características dos marcos têm de ser previamente conhecidas e guardadas na memória do robô.

A localização por reconhecimento de marcos é inerentemente intermitente, uma vez que os marcos se encontram espaçados no ambiente e, geralmente, um veículo só consegue reconhecer um marco de cada vez, quando se encontra bastante próximo deste. Para a navegação entre dois marcos é, então, necessário recorrer a outro método de localização (por exemplo, à odometria). Além disso, o veículo está condicionado a seguir percursos ao longo dos quais existam marcos. O não reconhecimento de apenas um dos marcos pode fazer com que o veículo se perca (NDC, 1998).

Os marcos dizem-se *naturais* quando já existiam no ambiente com uma qualquer finalidade diferente da de permitir a navegação. Os marcos *artificiais* são colocados no ambiente com o fim exclusivo de possibilitar a localização de veículos (Borenstein *et al.*, 1996; Borenstein *et al.*, 1997).

Os marcos podem ter formas geométricas simples (rectângulos, linhas ou círculos, por exemplo) e, no caso de serem artificiais, podem conter informação adicional (por exemplo sob a forma de códigos de barras). A maior parte dos marcos naturais utilizados com sistemas de visão por computador são linhas verticais (junções de portas e janelas, esquinas, candeeiros de iluminação pública), luzes no tecto ou componentes de cor RGB (Borenstein *et al.*, 1996; Wijk e Christensen, 2000; Clerentin *et al.*, 2002; Yuen e MacDonald, 2002; Kang e Jo, 2003).

Um sistema de medição de posição com marcos naturais possui os seguintes componentes básicos (Borenstein *et al.*, 1996):

- Um sensor para detectar os marcos e os contrastar com o seu fundo. Geralmente, o robô utiliza um sistema de visão por computador;
- Um método que permita comparar as características observadas pelo robô com um mapa de marcos conhecidos guardado em memória;
- Um método que permita calcular a posição e os erros de posição.

A chamada *visão global* consiste na utilização de câmaras fixas que identificam e localizam pontos característicos que formam um padrão num robô móvel, permitindo a sua localização (Borenstein *et al.*, 1996). Trata-se de um exemplo de localização remota que recorre a técnicas de reconhecimento de marcos naturais.

Borenstein *et al.* (1996) e Borenstein *et al.* (1997) tecem as seguintes considerações relativas à localização baseada no reconhecimento de marcos naturais:

- Os sistemas de navegação com marcos naturais oferecem flexibilidade e funcionam melhor em ambientes altamente estruturados (corredores, hospitais, etc.).

- Em muitos casos, os computadores existentes a bordo do robô não conseguem processar os algoritmos de reconhecimento de marcos naturais a uma velocidade suficientemente elevada para a localização em tempo real.

- A exactidão da localização depende da geometria do robô e dos marcos.

- O alcance efectivo é da ordem dos 10m.

- O apoio comercial às técnicas baseadas em marcos naturais é reduzido.

Os mesmos autores também referem outros aspectos que são comuns à localização com marcos naturais e à localização com marcos artificiais:

- Em geral, a exactidão com que a posição do robô é determinada diminui com o aumento da distância entre robô e marco. A navegação com marcos é bastante inexacta quando o robô se encontra longe do marco que nesse momento estiver a ser utilizado para a localização. Só se consegue um grau de exactidão mais elevado quando o robô está perto de um marco. A orientação do robô relativamente ao marco, expressa por um ângulo, é outro factor que determina esta exactidão. Existe normalmente uma gama de valores deste ângulo para os quais se consegue uma boa exactidão. Fora dessa gama, a exactidão diminui consideravelmente.

- As condições ambientais, por exemplo a iluminação, podem constituir um problema. Quando a visibilidade é fraca pode acontecer que os marcos não sejam reconhecidos ou que outros objectos com características parecidas sejam confundidos com marcos. Trata-se de um problema grave, pois pode resultar numa determinação da posição do robô completamente errada.

- A navegação por marcos requer que a posição e a orientação do robô sejam aproximadamente conhecidas à partida para que o robô só precise de procurar os marcos numa área limitada. Se a posição e a orientação forem desconhecidas, o robô tem de levar a cabo um processo de pesquisa demorado. Este processo de pesquisa pode falhar e produzir uma interpretação errónea dos objectos que estejam à vista. Torna-se assim patente que uma boa exactidão na localização por odometria constitui um pré-requisito para o sucesso da navegação com marcos.

- Tem de haver marcos disponíveis no ambiente de trabalho à volta do robô.

- É necessário manter uma base de dados sobre os marcos e as suas localizações no ambiente.

- Em diferentes projectos de investigação de reconhecimento de marcos obteve-se uma exactidão de posição da ordem de 5cm e uma exactidão de orientação da ordem de 1º.

A título de exemplo, Clerentin *et al.* (2002) obtiveram uma exactidão de posição de 13cm e uma exactidão de orientação de 3º com um sistema de localização que recorre ao reconhecimento de marcos naturais com uma câmara CCD e à medição de distâncias com um telémetro laser.

2.1.1.3 Correspondência de Mapas

A *correspondência de mapas (map-matching)* é um método de autolocalização de robôs móveis no qual um mapa local do ambiente, construído pelo próprio robô, é comparado com um mapa global guardado em memória. Se se estabelecer uma correspondência entre os dois mapas, o robô pode calcular a sua posição e orientação actuais no ambiente. A correspondência costuma ser baseada num processo de pesquisa.

Pode acontecer que o robô navegue num ambiente totalmente desconhecido, do qual não possui um mapa *a priori*. Nesse caso, o robô só consegue localizar-se relativamente às características que identifica no ambiente. Recorre-se habitualmente à sigla *SLAM (Simultaneous Localization and Mapping)* para designar o problema de um robô construir um mapa do ambiente, identificar marcos e, simultaneamente, localizar-

se relativamente a esses marcos (Di Marco *et al.*, 2000; Kleeman, 2003; Prasser e Wyeth, 2003; Tanaka *et al.*, 2003; Costa *et al.*, 2004). Saeedi *et al.* (2003) obtiveram uma exctidão de posição de cerca de 4cm, num ambiente desconhecido, com um veículo munido de um sistema de visão que actualiza as medições à frequência de 2,8Hz.

Para a *construção de mapas (map-building)* é necessário que no ambiente exista um número suficiente de características permanentes facilmente reconhecíveis e que o robô disponha de sensores adequados ao seu reconhecimento. Uma das desvantagens da navegação baseada na correspondência de mapas é que, para ser útil, o mapa construído pelo robô deve ser suficientemente exacto, e o rigor exigido costuma ser grande. É muito frequente recorrer a câmaras, integradas em sistemas de visão por computador. Também se utilizam outros sensores, tais como telémetros baseados em laser, infravermelhos ou ultra-sons (Everett, 1995; Borenstein *et al.*, 1996; Colon e Baudoin, 1996; Owen e Nehmzow, 1998; Kleeman, 2003; Lee *et al.*, 2003). Além disso, Borenstein *et al.* (1996) e Borenstein *et al.* (1997) referem que, na opinião de muitos investigadores, não é possível captar adequadamente todas as características relevantes de um ambiente real com um único tipo de sensor. Para ultrapassar esta dificuldade, é necessário combinar os dados provenientes de diversos tipos de sensores. A este processo chama-se *fusão sensorial.*

Quando se recorre a sistemas de visão, a construção do mapa do ambiente faz-se geralmente à custa da extracção de características[3] do ambiente detectadas em uma ou mais posições do robô. O robô deve possuir, à partida, uma estimativa da sua posição, obtida por odometria, para limitar a pesquisa de características a uma área menor (Borenstein *et al.*, 1996).

Em vez do uso de simples características, o ambiente pode ser descrito de uma forma mais abrangente recorrendo, por exemplo, a modelos geométricos bidimensionais ou tridimensionais das estruturas aí existentes (Borenstein *et al.*, 1996; Aider *et al.*, 2002). Para possibilitar a localização, as observações bidimensionais feitas pelos

[3] A correcta correspondência dessas características (umas com as outras) pode produzir informação sobre o movimento do robô (tanto de translação como de rotação) e também sobre a estrutura tridimensional do ambiente nas localizações onde as características se encontram. A trajectória do robô pode ser feita por integração do movimento incremental estimado como acontece na odometria. As características de um objecto detectadas numa determinada localização do robô transformam-se na referência relativa das seguintes localizações do robô. Quando as correspondências são correctamente estabelecidas, os métodos de visão podem produzir uma maior exactidão na estimativa da posição que a odometria ou os sistemas de navegação inercial (Borenstein *et al.*, 1996).

sensores do robô devem apreender as características do ambiente que possam corresponder ao modelo guardado em memória, com um mínimo de incerteza[4]. Borenstein *et al.* (1997) referem um sistema de localização por correspondência de modelos com uma exactidão de localização da ordem de 1cm a 10cm (posição) e 1º a 3º (orientação). O alcance efectivo desse sistema ronda os 10m.

Os diversos métodos de fazer a correspondência de mapas têm sido considerados muito lentos ou muito inexactos, pouco fiáveis e insuficientemente robustos para permitir a localização de robôs móveis em aplicações comerciais gerais, produzindo bons resultados apenas em ambientes laboratoriais bem estruturados e relativamente simples; além de requerem bastante processamento e capacidade sensorial, o tempo de processamento – que depende da resolução e dos algoritmos usados – pode chegar a ser demasiado longo para a navegação em tempo real (Borenstein, 1994, Borenstein e Feng, 1994, Borenstein e Feng, 1996b; Borenstein *et al.*, 1996; Borenstein *et al.*, 1997; Saeedi *et al.*, 2003). Todos estes motivos fazem desaconselhar o seu uso como base de um sistema fiável e de custo relativamente baixo para a localização em tempo real de robôs que se movem com velocidades de alguns metros por segundo.

2.1.2 Métodos que Requerem Preparação do Ambiente

Quando se recorre a estes métodos é necessário preparar o ambiente para a navegação, dotando-o de marcos ou balizas que aí são expressamente colocados para esse efeito.

2.1.2.1 Reconhecimento de Marcos Artificiais

A autolocalização baseada no reconhecimento de marcos artificiais é semelhante à que utiliza marcos naturais. A principal diferença é a colocação de marcos em posições conhecidas do ambiente no qual se move o robô, com o fim exclusivo de permitir a sua navegação.

[4] Leonard e Durrant-White (1992) referem que a localização por correspondência de modelos está relacionada com a visão baseada em modelos, cujo fim é *"reconhecer um objecto no ambiente com base num modelo a priori e na determinação da posição e da orientação desse objecto relativamente ao observador. O processo de reconhecimento é, fundamentalmente, uma pesquisa através de informação prévia; o objectivo do processamento visual é fornecer constrangimentos para guiar a pesquisa de forma a encontrar a orientação e a posição correctas do objecto o mais depressa possível"*. Os mesmos autores vêm a localização como *"uma tarefa de conseguir correspondência entre observações e um modelo a priori"*.

Os marcos artificiais são muito mais fáceis de detectar que os naturais. Muitos sistemas de reconhecimento de marcos artificiais recorrem à visão por computador mas há outras tecnologias disponíveis para o efeito. Eis alguns exemplos de marcos artificiais[5] (Berger e Kubitz, 1996; Borenstein *et al.*, 1996; Schreiber e Dickerson, 1996; Sena Esteves, 1996; Borenstein *et al.*, 1997; Lin e Tummala, 1997; Jang *et al.*, 2002):

- *Marcos ópticos*, tais como etiquetas retro-reflectoras iluminadas por uma fonte de luz (reconhecíveis por sistemas de *visão máquina*), etiquetas reflectoras com códigos de barras (reconhecíveis por *scanners laser*), ou ainda formas geométricas (facilmente reconhecíveis por sistemas de visão por computador). A forma e as dimensões exactas dos marcos, por serem conhecidas de antemão, podem servir para codificar informação. Estes marcos são de baixo custo.

- *Marcos magnéticos* embutidos no chão, detectados por um sensor de *Efeito Hall* colocado no robô, quando este passa por cima deles.

- *Transponders*[6] embutidos no chão, detectados por uma antena colocada no robô, quando este passa por cima deles.

- Superfícies que produzem um eco identificável quando atingidas por ultra-sons.

Muitas das características já referidas a propósito do reconhecimento de marcos naturais são comuns ao reconhecimento de marcos artificiais, nomeadamente as seguintes:

- A localização baseada no reconhecimento de marcos artificiais é inerentemente intermitente.

[5] Borenstein et al. (1996) fazem referência a *marcos contínuos*, como é o caso dos longos fios eléctricos enterrados no solo (AGVE, 2001a; AGVP, 2003c; EGEMIN, 2002e), os quais servem para guiar AGV's *(Automatic Guided Vehicles)*. Para o mesmo efeito também se recorre a fita reflectora (EGEMIN, 2002d) ou a tinta com ferrite em pó. No entanto, estes fios, fitas ou linhas destinam-se exclusivamente a guiar veículos e não permitem a sua localização.

[6] Um *transponder* é um receptor/emissor remotamente activado (Curtis, 1989; Teunon, 1992).

- O veículo tem de seguir percursos ao longo dos quais existam marcos.

- O não reconhecimento de apenas um marco pode fazer com que o veículo se perca.

- A navegação só costuma ser suficientemente exacta quando o robô se encontra perto de um marco e a orientação do robô relativamente ao marco se mantém dentro de uma gama bem definida.

- Existe uma grande sensibilidade às condições ambientais (em particular, à iluminação, nos sistemas de reconhecimento de marcos ópticos baseados em visão por computador).

- A posição e a orientação do robô devem ser aproximadamente conhecidas à partida (recorrendo, por exemplo à odometria) para que o robô só precise de procurar os marcos numa área limitada.

- Tem de haver marcos disponíveis no ambiente de trabalho em redor do robô.

- É necessário manter uma base de dados sobre os marcos e as suas localizações no ambiente.

- A exactidão de posição e a exactidão de orientação obtidas em diversos projectos de investigação de reconhecimento de marcos foram da ordem de 5cm e 1º, respectivamente.

A tecnologia dos AGVs *(Automatic Guided Vehicles)* guiados por fios (AGVE, 2001a; De Cecco, 2002; AGVP, 2003c) está provada, é bem conhecida e de fácil utilização, mas constitui uma solução pouco flexível e não permite localizar o veículo. Além disso, em muitas situações a instalação de fios no chão é difícil e/ou dispendiosa. Como alternativa, é frequente utilizar AGVs que navegam recorrendo à leitura periódica de marcos magnéticos (De Cecco, 2002) embutidos no solo, operação que também permite determinar a sua posição absoluta. Em AGVE (2001b) descreve-se um destes

sistemas, que utiliza um par de marcos por cada 5m ou 10m de percurso[7]. Para navegar entre marcos, os AGVs recorrem à odometria e a giroscópios.

2.1.2.2 Trilateração ou Triangulação com Balizas

Uma baliza é um dispositivo que, situado numa posição conhecida – mas não necessariamente fixa[8] – do espaço, possibilita a localização absoluta e contínua de um veículo ao longo de todos os trajectos que este possa percorrer mantendo uma comunicação em *linha de vista (line-of-sight)* com uma ou mais balizas[9]. Em geral, a localização com balizas pode fazer-se

- por *triangulação*, se for baseada na medição de ângulos[10] (Kuipers e Levitt, 1988; Sutherland e Thompson, 1994; Garulli e Vicino, 2001; Shimshoni, 2002; Hernández *et al.*, 2003; Briechle e Hanebeck, 2004);

- por *trilateração*, se for baseada na medição de distâncias ou de diferenças de distâncias[11] (Everett, 1995; Borenstein *et al.*, 1996; Zhao, 1997; Drane e Rizos, 1998; DoD, 2001);

- com base na medição simultânea de ângulos e distâncias (Di Marco *et al.*, 2000; DM, 2003; Prasser e Wyeth, 2003; SICK, 2003).

O *Global Positioning System (GPS)* da *Navstar* possibilita a autolocalização por trilateração, utilizando balizas que são satélites. Na situação da Figura 2.2, um robô que se move num plano recorre exclusivamente à medição de ângulos (λ_1, λ_2, λ_3) para determinar a sua posição (x_R, y_R) e a sua orientação (θ_R). As balizas estão representadas por círculos a vermelho.

---

[7] De acordo com esta fonte, a instalação dos marcos fica entre 4 e 8 vezes mais económica que a instalação de fios no chão. Mas os veículos com capacidade de reconhecer marcos magnéticos são mais dispendiosos que os guiados por fios (a diferença é de 5 a 10%), a guia é menos exacta e a instalação do sistema requer mais programação.

[8] Por exemplo, os satélites do *Global Positioning System (GPS)* são balizas que se movem em relação à Terra.

[9] Esta definição de baliza está relacionada com a de Navegação de Área ou *RNAV (Area Navigation)* dada pelo *ICAO (International Civil Aviation Organization): "Um método de navegação que permite a operação de um avião ao longo de qualquer percurso de voo desejado*" (EUROCONTROL, 1998).

[10] Os ângulos medidos com o fim de localizar um veículo podem ser os formados
a) pelo segmento de recta que une duas balizas com os segmentos de recta que unem essas balizas ao veículo;
b) pelos segmentos de recta que unem cada baliza ao veículo;
c) por um eixo de referência fixo no veículo com os segmentos de recta que unem o veículo a cada baliza.

[11] Em concreto, as distâncias do veículo a cada baliza ou as diferenças dessas distâncias.

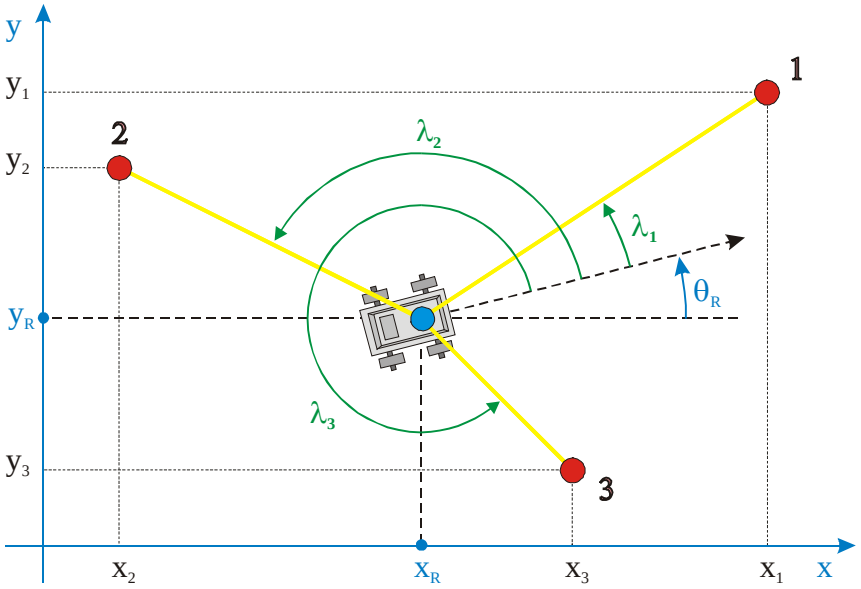


Figura 2.2: Triangulação na navegação a duas dimensões.

Uma baliza diz-se *activa* se possuir uma fonte de energia própria. Caso contrário, a baliza diz-se *passiva* (por exemplo, uma etiqueta reflectora). As balizas activas podem ser *emissoras* (por exemplo, um farol ou um satélite do GPS), *receptoras* (por exemplo, uma estação radiogoniométrica) ou *simultaneamente emissoras e receptoras* (por exemplo, uma estação de radar).

As balizas emissoras *não direccionais* emitem um sinal em todas as direcções, ou seja, possuem um padrão de transmissão omnidireccional (Figura 2.3a). Por isso também se lhes pode chamar balizas emissoras *omnidireccionais*. Se o sinal emitido possuir iguais características em todas as direcções, o padrão de transmissão da baliza é *isotrópico*. Caso contrário, o padrão de transmissão diz-se *anisotrópico*[12].

Para reduzir a potência inerente a uma transmissão omnidireccional, alguns sistemas utilizam balizas emissoras *direccionais*, cujos padrões de transmissão se restringem a uma porção do espaço, geralmente a um cone (Figura 2.3b). Muitos sistemas de localização a duas dimensões requerem que o veículo possa comunicar simultaneamente com um mínimo de três balizas. Na situação representada na Figura 2.4, isto só é possível numa pequena zona do plano.

[12] Um exemplo deste tipo de baliza é dado por Venet *et al.* (2002).

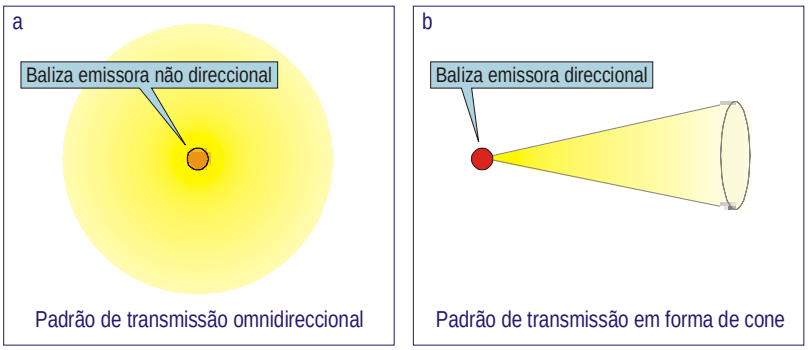


Figura 2.3: Padrões de transmissão de balizas activas emissoras: a) padrão de transmissão omnidireccional; b) padrão de transmissão em forma de cone.

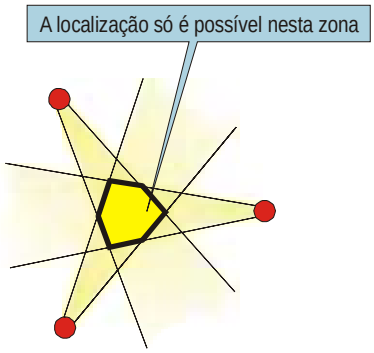


Figura 2.4: Zona na qual é possível fazer a localização.

À semelhança do que fazem Borenstein *et al.* (1996) – e ao contrário de muitos autores – neste trabalho distinguem-se os termos *marco (landmark)* e *baliza (beacon).* Considera-se que uma determinada característica do ambiente – um marcador (por exemplo uma etiqueta retro-reflectora com um código de barras) – recebe a designação de marco ou de baliza de acordo com o método utilizado para a determinação da posição:

- Se o robô determinar a sua própria posição a partir do reconhecimento de um único marcador seguido da atribuição da posição desse marcador ao robô, então esse marcador é um marco. É o que acontece, por exemplo, na *navegação por pontos (point navigation* ou *signpost navigation).* Neste tipo de navegação, a determinação de posição é intermitente, uma vez que só se faz quando o veículo se encontra junto de um marco. Além disso, o veículo está limitado a seguir percursos que passem pelos marcos.

- Quando o robô recorre à triangulação, à trilateração ou à medição simultânea de distâncias e ângulos, cada um dos vários marcadores utilizados simultaneamente é uma baliza[13]. A localização com balizas é contínua ao longo de todos os trajectos que o veículo possa percorrer mantendo uma comunicação em linha de vista com uma ou mais balizas. Desta forma, o veículo não é obrigado a seguir percursos que passem pelas balizas.

Podem apontar-se mais algumas diferenças importantes entre marcos e balizas (Borenstein *et al.*, 1996):

- A posição de um marco é geralmente importante. Pelo contrário, a posição de uma baliza é – em si mesma – irrelevante (embora costume ser cuidadosamente escolhida).

- A distância máxima entre um robô e um marco é consideravelmente mais pequena do que a que pode existir entre o robô e uma baliza activa.

- O reconhecimento de marcos requer geralmente mais processamento do que a localização com balizas activas (sobretudo nos casos do reconhecimento de marcos naturais ou do reconhecimento de marcos artificiais recorrendo a sistemas de visão por computador).

Os sistemas de localização com balizas podem classificar-se de acordo com a tecnologia utilizada nas comunicações entre um veículo e cada baliza. Eis alguns exemplos:

- Sistemas ópticos baseados em *laser*;

- Sistemas de ultra-sons que utilizam *transponders* como balizas;

[13] Num sistema de localização baseado no reconhecimento de marcos é possível – pelo menos em princípio – determinar a posição actual de um robô por triangulação ou por trilateração, desde que o robô aviste um número suficiente de marcos (os quais, nesse caso, passam a actuar como balizas). No entanto, muitos sistemas de reconhecimento de marcos só possuem a capacidade de reconhecer um marco de cada vez (ou seja, em cada ponto do plano de navegação, o sistema só consegue – no máximo – reconhecer um marco). Mesmo que não houvesse essa limitação, com muitos sistemas não é viável reconhecer vários marcos e determinar a posição do robô em tempo real.

- Sistemas de radiofrequência, tais como as várias versões do *LORAN (Long Range Navigation)* ou o GPS.

A trilateração com *transponders* de ultra-sons constitui uma solução de média a alta exactidão e baixo custo ao problema da determinação da posição de robôs móveis (Everett, 1995; Borenstein *et al.*, 1996). Devido ao facto de os ultra-sons possuírem um alcance relativamente reduzido (tipicamente, poucas dezenas de metros), estes sistemas só são adequados à operação em pequenas áreas de trabalho com poucos obstáculos que interfiram com a propagação das ondas de ultra-sons. Não são convenientes para aplicações em instalações grandes, com várias dependências, por causa da significativa complexidade inerente à instalação de várias balizas ligadas em rede.

Na Tabela 2.2 encontram-se algumas características de sistemas de trilateração que utilizam balizas de ultra-sons.

Tabela 2.2: Características de sistemas de trilateração com balizas de ultra-sons (Everett, 1995; Borenstein *et al.*, 1996).

Características de Sistemas de Trilateração com Balizas de Ultra-sons						
Sistema	**No veículo**	**Balizas**	**Exactidão de Posicionamento**	**Alcance**	**Custo**	**Diversos**
Produzido por IS Robotics, Inc.	Transdutor receptor de ultra-sons	2 emissores de ultra-sons colocados a uma distância conhecida (tipicamente 2,28m) um do outro	12,7 mm	Distâncias compreendidas num quadrado de 9,1m de lado	$10000 (sistema)	-
Desenvolvido pela Universidade de Tulane	Emissor de ultra-sons	5 receptores de ultra-sons montados no tecto	0,25 mm	Distâncias compreendidas num quadrado de 2,7m x 3,7m	-	Frequência de actualização das medidas: 100hz Pode ser usado em recintos maiores se forem usados mais receptores.

Os sistemas que utilizam balizas de radiofrequência recorrem geralmente à trilateração e podem ser terrestres (Tabela 2.3) ou de infra-estrutura espacial. Podem possuir alcances muito elevados – o sistema GPS até oferece uma cobertura global. No entanto, nenhum dos sistemas de radiofrequência se pode usar de um modo fiável em interiores (Borenstein *et al.*, 1996; Borenstein *et al.*, 1997). Há algumas hipóteses de êxito se for possível manter uma linha de vista entre as balizas e o veículo. Mas, nesse caso, os sistemas ópticos que usam triangulação são geralmente mais económicos que os de radiofrequência (Borenstein *et al.*, 1997).

Tabela 2.3: Características de sistemas terrestres de trilateração com balizas de radiofrequência (Everett, 1995; Borenstein et. al., 1996).

Características de Sistemas Terrestres de Trilateração com Balizas de Radiofrequência						
Sistema	**No veículo**	**Balizas**	**Exactidão de Posicionamento**	**Alcance**	**Custo**	**Diversos**
LORAN-C (MIT)	Receptor *LORAN-C*	Torres de 400m de altura que emitem sinais de 5MW com uma portadora de 100kHz	cerca de 100m (melhor caso)	1000 milhas	- (Apenas o custo do receptor)	Possui 50 emissores situados ao longo de todas as águas costeiras dos Estados Unidos e partes do Atlântico Norte, Pacífico Norte e Mediterrâneo.
Mini-Ranger Falcon (Motorola)	Interrogador de *transponders* (emissor/ receptor)	2, 3, 4 ou 16 *transponders* a operar na banda C (5410-5890MHz), cada um dos quais responde apenas a uma única interrogação codificada	2m (provável)	100m a 75km	$75000 a $100000 (sistema completo)	Pode ser utilizado por um número máximo de 20 veículos (*time-sharing* de 50ms por utilizador, no máximo). Resolução: decímetros. Frequência de actualização das medidas: 1Hz
Precision Location (Precision Technology, Inc.)	Emissor de um sinal sinusoidal contínuo de 58MHz	Várias antenas receptoras	1 a 10cm	-	$200000 a $400000, dependendo do número de receptores	Não requer comunicações em linha de vista. Não é adequado à navegação em interiores. Apresenta dificuldades na localização simultânea de vários veículos.

O LORAN-C serve bastantes regiões do planeta e os seus utilizadores apenas têm de adquirir um receptor adequado, mas exibe uma exactidão de posicionamento da ordem dos 100m, insuficiente para a generalidade das aplicações da robótica móvel. Os sistemas terrestres de maior exactidão (metros ou centímetros) são geralmente muito dispendiosos.

O aparecimento do GPS revolucionou a localização em ambientes exteriores (Hurn, 1989) e o seu domínio aplicacional encontra-se em franca expansão. Por isso, este sistema será alvo de um estudo mais aprofundado, nos próximos parágrafos.

A medição da posição de um receptor GPS faz-se a partir da medição das distâncias entre a antena desse receptor e as antenas de vários satélites emissores (pelo menos três, para se poder obter as duas coordenadas de posição horizontal). A frequência à qual são actualizadas as medidas de posição é de 1 a 20Hz (DoD, 2001).

Para medir a distância de um receptor GPS a um satélite recorre-se ao seguinte princípio: um código binário do tipo *PRN (Pseudo-Random Noise)* é simultaneamente gerado no satélite e no receptor, que se encontram sincronizados. O código gerado no satélite é usado para modular uma portadora e o sinal obtido é transmitido. No receptor

GPS compara-se o desfasamento existente entre o código contido no sinal recebido e o código gerado internamente. Este desfasamento é proporcional ao tempo que o sinal transmitido demora a atingir a antena do receptor. Por sua vez, este tempo é proporcional à distância percorrida pelo sinal entre as antenas do satélite e do receptor, que é calculada partindo do princípio que o sinal se propaga a uma velocidade constante e conhecida. Na prática, a medida da distância é apenas aproximada[14] e costuma chamar-se *pseudo-distância.* É neste princípio que se baseia o *Standard Positioning Service (SPS),* que o GPS disponibiliza para os utilizadores civis, e que garante uma *exactidão de posicionamento horizontal*[15] de alguns metros. É possível obter uma exactidão de posicionamento da ordem dos poucos centímetros (ou até dos milímetros, em condições especiais) recorrendo a métodos baseados não só na medição dos desfasamentos entre códigos PRN, mas também na medição da fase da portadora dos sinais que contêm os códigos PRN gerados nos satélites.

Os erros de posicionamento dependem não só dos erros de medição das pseudo-distâncias mas também das posições dos satélites relativamente ao receptor GPS. Os erros de medição são multiplicados por um factor que pode variar entre cerca de 1,5 (para satélites dispersos pelo céu) e 5 ou mais (se os satélites estiverem agrupados) (Bretz, 2000).

O modo de posicionamento mais simples é o *Single Point Positioning (SPP),* no qual se utiliza um único receptor GPS colocado no ponto cuja posição se pretende medir (Rizos e Satirapod, 2001). Os métodos *Differential GPS (DGPS)* de tempo real requerem a existência de (pelo menos) dois receptores GPS com a capacidade de comunicar entre si. Um dos receptores (a *estação base*) é colocado numa posição fixa conhecida e emite em tempo real, para um ou mais receptores GPS móveis, um sinal com correcções relativas às fases dos códigos PRN contidos nos sinais recebidos do espaço. O *Real-Time Kinematic (RTK) GPS* é um método diferencial no qual a estação base emite correcções relativas não só às fases dos códigos PRN, mas também à fase da portadora desses códigos.

[14] Como fontes importantes de erros de medição podem apontar-se, por exemplo, o facto de o receptor não se encontrar perfeitamente sincronizado com o satélite e também o facto de o sinal sofrer atrasos variáveis na ionosfera e na troposfera.

[15] *"Exactidão de Posicionamento Horizontal: Definida como sendo a diferença estatística, a uma probabilidade de 95%, entre as medidas de posição horizontal e um ponto de referência observado para qualquer ponto dentro do volume de serviço num qualquer intervalo de 24 horas."* (SPS-PS, 2001).

Uma das principais limitações do GPS é o facto de os sinais recebidos à superfície da Terra serem muito fracos. Podem ser facilmente bloqueados por montes e prédios altos ou fortemente atenuados por paredes e outros obstáculos. Isto faz com que a localização em exteriores seja intermitente na presença de alguns obstáculos e extremamente difícil em locais rodeados de edifícios muito altos. A localização em interiores é também muito difícil. Para o sinal destinado aos utilizadores civis, o Departamento de Defesa dos Estados Unidos garante uma potência mínima à superfície da Terra de apenas -160dBW (10^{-16}W) [16]. A potência dos sinais em interiores é inferior em 20 ou 30dB (Diggelen e Abraham, 2001), sendo necessário recorrer a sofisticados receptores de alta sensibilidade. Com um desses receptores, Diggelen e Abraham (2001) obtiveram uma exactidão de posição entre 20 e 24m em interiores e de 25m no exterior, numa rua rodeada de arranha-céus. Mesmo em exteriores pouco obstruídos, o recurso exclusivo ao GPS não constitui uma solução adequada para medir em tempo real a posição de robôs móveis com uma exactidão da ordem dos milímetros:

- Recorrendo ao SPP com um receptor GPS de baixo custo, que determine em tempo real a sua posição utilizando apenas o SPS, a exactidão de posicionamento horizontal actualmente garantida pelo Departamento de Defesa dos Estados Unidos é de 13m[17], no melhor dos casos (SPS-PS, 2001). Este valor é para sinais no espaço, ou seja, não inclui os efeitos devidos aos atrasos dos sinais na ionosfera e na troposfera, às interferências (nomeadamente à provocada pela propagação multitrajectos) e às limitações do próprio receptor GPS.

- Recorrendo ao DGPS apenas com correcções de fase dos códigos PRN pode obter-se, em tempo real, uma exactidão de posicionamento da ordem de 1m (Everett, 1995; Trimble, 2000; Grejner-Brzezinska, 2002; Trimble, 2002).

[16] Para um sinal medido à saída de uma antena receptora de 3 dBi linearmente polarizada, colocada próxima do solo, quando o satélite emissor se encontra acima de um ângulo de elevação de 5º, assumindo perdas por atenuação atmosférica de 2 dB (ICD-200, 2000; SPS-PS, 2001). Na realidade, esta potência é um pouco superior: cerca de -157,6dBW (Braasch e Dierendonck, 1999). Com os novos satélites IIF prevê-se que a potência mínima à superfície da Terra para este sinal será de -157,5dBW (Fisher e Ghassemi, 1999) ou de -157,7dBW (Fontana et al., 2001b). Estão previstos dois novos sinais destinados a utilizadores civis, cujas potências à superfície da Terra deverão ser de -160dBW e -154dBW. Devem estar disponíveis por volta de 2011 e 2015, respectivamente (Dierendonck, 2001; Fontana et al., 2001a; Fontana et al., 2001b).

$$P_{dB} = 10\log P_W \quad \Rightarrow \quad -160 = 10\log P_W \quad \Rightarrow \quad P_W = 10^{-16}$$

[17] Rizos e Satirapod (2001) e Satirapod *et al.* (2001) obtiveram, mediante ensaios, um valor de cerca de 7m.

- É necessário recorrer ao RTK GPS para se obter, em tempo real, uma exactidão de posicionamento horizontal da ordem de 1cm a 2cm, podendo a frequência de actualização das medidas chegar a 20Hz (Everett, 1995; Trimble, 2000; Grejner-Brzezinska, 2002; Jong, 2002; Trimble, 2002). No entanto, o preço dos receptores é demasiado elevado para a generalidade das aplicações da robótica móvel.

- Para se obter uma exactidão de posicionamento da ordem dos milímetros é necessário utilizar métodos diferenciais baseados na medição da fase da portadora dos sinais emitidos pelos satélites, realizar medições durante longos períodos de tempo (horas) no ponto cuja posição se pretende determinar e fazer um pós-processamento significativo das medidas obtidas (Kelly, 1996), o que é incompatível com aplicações em tempo real.

- As interferências devidas à propagação multitrajectos dos sinais emitidos pelos satélites produzem erros quer na fase dos códigos PRN quer na fase da portadora. Os métodos DGPS não permitem reduzir estes erros (Bretz, 2000), uma vez que dificilmente se verifica que um sinal reflectido atinja simultaneamente a estação base e os receptores GPS móveis, sobretudo se a estação base se encontrar longe dos outros receptores.

  a) Os erros produzidos pelos sinais reflectidos nas pseudo-distâncias calculadas a partir da fase dos códigos PRN raramente atingem o seu limite máximo, que é de 147m. No entanto, têm sido medidos erros superiores a 100m com receptores GPS estacionários colocados perto de arranha-céus. Para receptores GPS afastados de objectos grandes são muito comuns os erros da ordem dos 10m (Braasch e Dierendonck, 1999).

     Recorrendo a tecnologias de rejeição de interferências por propagação multitrajectos e a métodos diferenciais baseados na medição da fase dos códigos PRN, é possível atingir 30cm de exactidão de posicionamento horizontal – mas os receptores de mais baixo custo não costumam possuir nenhuma protecção contra interferências por propagação multitrajectos (Grejner-Brzezinska, 2002).

  b) Os erros produzidos pelos sinais reflectidos na fase da portadora podem atingir os 4,8cm. Tipicamente, são inferiores a 1cm. Há muito pouco trabalho publicado sobre a redução dos erros da fase da portadora devidos à propagação multitrajectos (Braasch e Dierendonck, 1999; Grejner-Brzezinska, 2002).

- O tempo necessário para um receptor GPS determinar a sua posição pela primeira vez depois de ser ligado, ou depois de ter perdido o sincronismo com os satélites que estava a utilizar para se posicionar, pode variar entre 30s e 15m (Reed e James, 1997; Trimble, 2000; Trimble, 2002).

- O Departamento de Defesa dos Estados Unidos pode proceder à degradação intencional da qualidade dos sinais emitidos pelos satélites. Isto foi feito desde 1990 até 1 de Maio de 2000, mediante um processo chamado *Selective Availability*, que afectou os sinais emitidos por todos os satélites, limitando a 100m a exactidão de posicionamento horizontal garantida para o SPP com um receptor GPS de baixo custo, que determine em tempo real a sua posição utilizando apenas o SPS utilizando apenas o SPS (Torres e Lima, 1996; Enge e Misra, 1999). Os Estados Unidos não tencionam voltar a utilizar este processo de degradação global, mas encontram-se em estudo métodos que permitam recusar capacidades do sistema localmente (μ-BLOX, 2001).

Os sistemas ópticos de localização são muito exactos, de custo relativamente baixo e funcionam de modo fiável em interiores. São geralmente mais exactos e mais económicos que os sistemas de radiofrequência. Os alcances da ordem das dezenas ou centenas de metros são suficientes em inúmeras aplicações. No entanto, a instalação e manutenção das balizas pode ser complexa e dispendiosa.

Os exemplos apresentados na Tabela 2.4 tornam bem patente que existe actualmente a tecnologia necessária à implementação de sistemas ópticos de trilateração e de triangulação. Por exemplo, quer o *Laser Scanner 4-2.0* da *Danaher Motion* (DM, 2003) quer o *scanner* laser do sistema de navegação *NAV 200* da *SICK* (SICK, 2003) efectuam medições de ângulos e de distâncias. O primeiro possui um alcance entre 1 e 70m, uma resolução angular de 0,057º (1mrad) e roda à frequência de 6Hz (38rad/s). O segundo tem um alcance de 1,2 a 30m e roda à frequência de 8Hz. Permite obter, quando integrado no sistema NAV 200, uma exactidão de posicionamento entre 4 e 25mm e uma exactidão de orientação de 0,1º.

Tabela 2.4: Características de sistemas ópticos de localização com balizas (Everett, 1995; Borenstein *et al.*, 1996; Borenstein *et al.*, 1997; NDC, 1998; SICK, 2003).

Características de Sistemas Ópticos de Localização com Balizas

Sistema	No veículo	Balizas	Exactidão (Posição)	Exactidão (Orient.)	Freq. Actualiz. Medidas	Alcance	Custo e Diversos
Do robô *Hilare (Lab. d'Automat. et d'Analyse des Systemes)*	2 emissores/ detectores *near-IR* num mastro rotativo	3 reflectores passivos	-	-	-	-	-
NAMCO LASERNET (Namco Controls Corp.)	Espelho rotativo que difunde um feixe laser *near-IR*	Alvos retro-reflectores de dimensões conhecidas	-	S. Analógica: ±1% (resolução de 0,1º) Saída RS-232: ±0,05% (resolução de 0,006º)	20Hz	15m	$3400
Com sensor *LaserNav (Denning Branch Int. Robotics)*	*LaserNav:* Scanner laser rotativo com uma exactidão de 0,03º a 600rpm	*Transponders* electrónicos ou reflectores	-	-	10Hz	183m com *transponders* 30,5m com reflectores	S*canner* distingue até 32 balizas numa só passagem
TRC Beacon Navigation System (Transitions Research Corp., posteriorm. *Helpmate Robotics* e *Pyxis Corp.*)	*Scanner* laser rotativo (60rpm) numa caixa cúbica com 100mm de lado	4 retro-reflectores passivos nos vértices de um recinto quadrado	- (resolução de 120mm)	- (resolução de 0,125º)	1Hz	Como não reconhece novas balizas, a operação está limitada a um recinto quadrado com 24,4m de lado	$11000 Ao fim de 15s iniciais, só duas balizas devem permanecer visíveis
Com *rangefinder ROBOSENSE (Siman Sensors & Intelligent Mach., Ltd.)*	*Rangefinder* laser rotativo	Alvos retro-reflectores	20mm	Melhor que 0,17º	10 a 40Hz	0,3 a 30m	$12800 (1un.) $7630 (>3 un.) Fornece coordenadas x e y, orientação e um nível de confiança
CONAC original *(MTI Research, Inc.)*	*STROAB*: Emissor laser rotativo (3000rpm)	*NOADs*: Receptores laser ligados a um PC (que escolhe dinamicamente o melhor conjunto de 3 *NOAD*'s)	Interiores: ±1,3mm Exteriores: ±5mm	Interiores e Exteriores: ±0,05º	25Hz	>100m	$6000 Não tem capacidade de recorrer à navegação estimada
CONAC segunda versão *(MTI Research, Inc.)*	*STROAB*: Sensor omnidirec_ cional que detecta várias balizas ao mesmo tempo	*2 NOADs*: Emissores laser rotativos sincronizados	Exteriores: ±1,3mm	Exteriores: ±0,05º	20Hz	250m	$4000 (1 *STROAB* + 3 *NOADs*)
Odyssey (Spatial Positioning Systems, Inc.)	Receptor óptico	2 ou mais transmissores laser	±1mm + 100ppm	-	5Hz	150m (exteriores) 75m (interiores)	$90000 Fornece 3 coordenadas de posição
Do cortador de relva autónomo CALMAN	Scanner laser rotativo	Tiras retro-reflectoras verticais	Erro de posição <2cm para velocidades até 0,3m/s	-	-	-	-
Lazerway (NDC)	Scanner laser rotativo	Tiras reflectoras	±2mm	-	20Hz	-	Para AGVs c/ velocidades até 1m/s
Nav 200 (SICK AG)	Scanner laser rotativo	Tiras reflectoras	4 a 25mm	0,1º	8Hz	1,2 a 30m	-

Em resumo, a localização com balizas possui as seguintes características (Borenstein, 1994; Borenstein e Feng, 1996b; Borenstein *et al.*, 1996; Borenstein *et al.*, 1997; AGVE, 2000; AGVP, 2003b; EGEMIN, 2002b):

- A detecção das balizas activas é muito fiável.
- A informação obtida sobre a posição e a orientação é muito exacta, sobretudo nos sistemas ópticos.
- O processamento da informação é mínimo (em particular, muito menor que no caso do reconhecimento de marcos). Alguns sistemas permitem taxas de actualização das medidas muito elevadas, pelo que a localização se pode considerar praticamente contínua.
- A distância máxima eficaz entre um veículo e uma baliza activa é consideravelmente superior à existente entre um veículo e um marco.
- O ambiente tem de ser modificado e alguns sistemas requerem tomadas eléctricas ou manutenção de baterias para as balizas. Em alguns sistemas as balizas devem estar sincronizadas. A existência de balizas pode causar uma certa perturbação em alguns ambientes.
- É necessário montar as balizas de uma forma exacta para que seja também exacta a informação sobre a posição.
- O custo de instalação e manutenção é elevado. Os sistemas de localização com balizas activas são muito mais dispendiosos que outros sistemas de localização (por exemplo os que se baseiam no reconhecimento de pastilhas magnéticas).
- Tem de se manter uma *linha de vista* entre o robô e um número mínimo de balizas (esse número depende do método utilizado e da aplicação concreta).
- A localização com balizas pode ser difícil (ou mesmo impossível) de implementar em certos ambientes, nomeadamente corredores estreitos ou zonas com muitos obstáculos.

2.2 Medição de Posição e Orientação Relativas

Em muitas situações reais, não é possível medir em todos os instantes a posição e orientação absolutas de um veículo. É o que acontece, por exemplo, quando se navega com marcos ou quando se recorre à triangulação e o veículo perde durante algum tempo a linha de vista com as balizas. Se a navegação se baseasse exclusivamente nessas referências externas tornar-se-ia impossível localizar o veículo em todos os instantes. O que nessas circunstâncias se faz é calcular a posição e a orientação actuais do veículo mediante a actualização de uma posição e uma orientação medidas anteriormente. Para se fazer essa actualização é necessário conhecer os seguintes parâmetros (Moody, 1971; MN, 1989):

- distância percorrida pelo veículo desde a antiga posição (ponto de partida);
- direcção do movimento.

Por sua vez, a distância percorrida pode determinar-se indirectamente, através das medidas de:

- velocidade do veículo;
- período de tempo decorrido entre o ponto de partida e a posição actual.

A posição e a orientação – num determinado referencial conhecido – calculadas desta maneira designam-se *posição relativa* e *orientação relativa* Ao processo de as determinar chama-se *localização relativa* (Aider *et al.*, 2002; De Cecco, 2002; Venet *et al.*, 2002; Hernández *et al.*, 2003). É este tipo de localização que caracteriza a *navegação estimada (dead-reckoning)* (MN, 1989; Everett, 1995; Soares e Restivo, 1997; Cauchois *et al.*, 2002; Costa *et al.*, 2004; Gning e Bonnifait, 2004).

Mesmo quando se utilizam outras formas de navegação, considera-se geralmente que a navegação estimada é essencial por questões de segurança. As estimativas de posição e de orientação que esta fornece continuamente são muito úteis na avaliação de todas as outras informações disponíveis sobre a navegação.

Todos os métodos que seguidamente se descrevem são usados na *autolocalização relativa* de robôs móveis e não requerem a preparação do ambiente para a navegação.

2.2.1 Odometria

Na navegação de robôs móveis chama-se *odometria* à determinação da distância percorrida por um veículo, da sua velocidade e da direcção do movimento a partir da medição da rotação das suas rodas. Os sensores mais utilizados para o efeito são potenciómetros, *resolvers* ou *encoders* ópticos montados no veio das rodas do robô (Everett, 1995; Borenstein *et al.,* 1996; Gu *et al.,* 2002).

A odometria baseia-se na integração ao longo do tempo de informação incremental sobre o movimento. Para se obter a posição integra-se a velocidade, em módulo e direcção (Frappier *et al.,* 1992), partindo do princípio que o deslocamento linear de cada roda do veículo em relação ao pavimento se pode deduzir da sua rotação. Na prática, isto só é aproximadamente verdade. Como faz notar Crowley (1995), *"a execução perfeita de uma trajectória utilizando apenas realimentação odométrica não é possível para um robô móvel"*. Há diversas fontes de erros de localização inerentes à odometria (McKerrow, 1991; Turennout e Honderd, 1992; Crowley, 1995; Borenstein e Feng, 1996; Borenstein *et al.,* 1996; Borenstein *et al.,* 1997; Di Marco *et al.,* 2000; Aider *et al.,* 2002; Gu *et al.,* 2002; Hernández *et al.,* 2003), que podem ser agrupadas em cinco categorias:

1. Imperfeições do robô que produzem erros sistemáticos de localização:

    - Diferentes diâmetros das rodas;
    - Diâmetro médio real das rodas diferente do nominal;
    - Distância real entre eixos diferente da nominal;
    - Rodas mal alinhadas;
    - Deformação dos pneus;
    - Variação do raio das rodas com a carga do veículo.

2. Imperfeições do robô que produzem erros não sistemáticos de localização:

    - Resolução finita dos *encoders;*

3. Interacção do chão com as rodas do robô, que produz erros não sistemáticos de localização e não pode ser modelada na perfeição:

    - Deslocação sobre pavimentos irregulares;
    - Deslocação sobre objectos inesperados no chão;
    - Resvalamento das rodas devido a:
        - pavimentos escorregadios;
        - aceleração excessiva;
        - curvar rapidamente (derrapagem);
        - forças externas (interacção com corpos exteriores);
        - forças internas (rodas tipo pé de mesa);
        - falta de contacto pontual entre as rodas e o pavimento.

4. Ruído e erros nos sinais provenientes dos sensores;

5. Aproximações introduzidas pelas equações usadas na odometria, que aproximam um movimento arbitrário por uma série de pequenos segmentos de recta.

Os erros sistemáticos são particularmente prejudiciais pelo facto de se acumularem constantemente. Quando a navegação se faz sobre superfícies ásperas e irregulares, os erros não sistemáticos podem ser dominantes. Mas quando um veículo se move sobre superfícies regulares e macias, o efeito dos erros sistemáticos é muito maior que o dos erros não sistemáticos.

A acumulação de erros é a principal desvantagem inerente à odometria (Di Marco *et al.,* 2000; Aider *et al.,* 2002; Cauchois *et al.,* 2002; De Cecco, 2002; Gu *et al.,* 2002; Venet *et al.,* 2002; Hernández *et al.,* 2003). Em particular, como referem Borenstein *et al.* (1997), a acumulação de erros de orientação conduz a grandes erros de posição, os quais têm um aumento proporcional à distância percorrida pelo robô. Tipicamente, a estimativa interna da posição do robô está completamente errada ao fim de um percurso de 10m (Borenstein, 1994; Borenstein e Feng, 1995). Apesar disso, a

odometria é o método mais usado para determinar a posição instantânea de robôs móveis (Borenstein e Feng, 1996b; Borenstein *et al.*, 1997; Hernández *et al.*, 2003).

Borenstein *et al.* (1996) referem que a odometria se utiliza em praticamente todos os robôs móveis pelas seguintes razões:

- Pode fazer-se a fusão dos dados provenientes da odometria com medidas de posição absoluta para obter estimativas de posição melhores e mais fiáveis (Di Marco *et al.*, 2000; Kleeman, 2003).

- Pode usar-se odometria entre duas actualizações de posição absoluta obtidas por reconhecimento de marcos. Para uma dada exactidão de medição de posição pretendida, o aumento da exactidão na odometria permite a redução do número de actualizações de posição absoluta (Borenstein e Feng, 1996b). Assim, podem usar-se menos marcos ao longo do percurso efectuado (o que reduz o custo da instalação). Além disso, poupa-se tempo. De facto, tanto o reconhecimento de marcos como a correspondência de mapas são processos lentos quando comparados com a velocidade à qual os robôs se movem. A odometria é muito mais rápida (e muito mais simples) que esses métodos (Turennout e Honderd, 1992; Crowley, 1995; Borenstein e Feng, 1996b; Di Marco *et al.*, 2000; Aider *et al.*, 2002).

- Muitos algoritmos baseados na correspondência de mapas de marcos partem do princípio que o robô pode manter a sua posição suficientemente bem, de modo a permitir-lhe procurar marcos numa área limitada e comparar características nessa área limitada. Consegue-se, assim, uma redução no tempo de processamento e também a melhoria da correcção na correspondência.

- Em alguns casos, a odometria é a única fonte de informação disponível para a navegação. É o que acontece, por exemplo, quando não há referências externas, quando as circunstâncias impedem a colocação ou selecção de marcos no ambiente, ou quando outro subsistema sensorial cessa de fornecer dados utilizáveis (Gu *et al.*, 2002; Gning e Bonnifait, 2004).

Borenstein (1994) e Borenstein e Feng (1995, 1996b) apresentam contribuições para a melhoria da exactidão da odometria. Há sistemas com grande imunidade aos erros sistemáticos. Pela sua natureza, estes são mais fáceis de corrigir – por um processo de calibração – que os erros não sistemáticos. A título de exemplo, um robô *LabMate* com um sistema de odometria calibrado pelo processo *UMBmark (University of Michigan Benchmark Test)* (Borenstein e Feng, 1994; Borenstein e Feng, 1995) descreveu um percurso quadrado com 4m de lado. Submeteu-se ao mesmo ensaio um robô *OmniMate*. A exactidão de posição e a exactidão de orientação obtidas em cada caso estão indicadas na Tabela 2.5 (Borenstein *et al.*, 1997).

Tabela 2.5: Exactidão de posição e exactidão de orientação obtidas na localização por odometria de dois robôs móveis.

LabMate		
	Exactidão na posição	**Exactidão na orientação**
Chão liso	30mm	1º-2º
10 solavancos	500mm	8º

OmniMate		
	Exactidão na posição	**Exactidão na orientação**
Chão liso	~20mm	<1º
10 solavancos	~40mm	<1º

Com um sistema de odometria baseado em *encoders* aplicados às rodas de um robô móvel, em laboratório, Turennout e Honderd (1992) obtiveram, ao fim de um percurso rectilíneo de 30m, uma exactidão de 1cm na direcção longitudinal e de 2cm a 3cm na direcção perpendicular ao movimento.

Em resumo, como principais vantagens dos sistemas de navegação que recorrem à odometria podem apontar-se:

- Simplicidade;
- Autonomia total (a navegação não depende de referências externas);
- Boa exactidão a curto prazo;
- Baixo custo;
- São possíveis taxas de amostragem muito elevadas que permitem operação em tempo real[18].

[18] Os *encoders* produzem realimentações com atrasos da ordem das centenas de nanosegundos (Crowley, 1995).

2.2.2 Utilização de Sensores Doppler

Os sistemas de navegação com sensores *Doppler* (Everett, 1995; Borenstein *et al.*, 1996) são habitualmente usados em aplicações náuticas e aeronáuticas para efectuar a medição da velocidade em relação à Terra, eliminando os erros de odometria introduzidos por correntes marítimas ou de ar desconhecidas.

O princípio de funcionamento dos sensores baseia-se no efeito *Doppler*, que consiste na mudança de frequência que se observa quando a energia radiada por um emissor é reflectida por uma superfície que se move em relação a esse emissor, como se ilustra na Figura 2.5. Para calcular a velocidade do veículo recorre-se às seguintes expressões:

$$V_A = \frac{V_D}{\cos\alpha} = \frac{cF_D}{2F_0\cos\alpha} \tag{2.1}$$

$$F_D = F_R - F_0 \tag{2.2}$$

em que:

V_A - velocidade do pavimento em relação ao veículo, ao longo do percurso;
V_D - velocidade *Doppler* medida;
α - ângulo de inclinação;
c - velocidade da luz;
F_0 - frequência transmitida;
F_R - frequência recebida (Barney, 1988).

A posição do veículo pode obter-se integrando a velocidade em ordem ao tempo.

Por razões económicas, e uma vez que a probabilidade de deslocamento transversal é reduzida, na maior parte das aplicações de robótica recorre-se a um único sensor *Doppler* para medir a velocidade do pavimento em relação ao robô, na direcção do movimento. À semelhança da odometria, este método não recorre a referências externas.

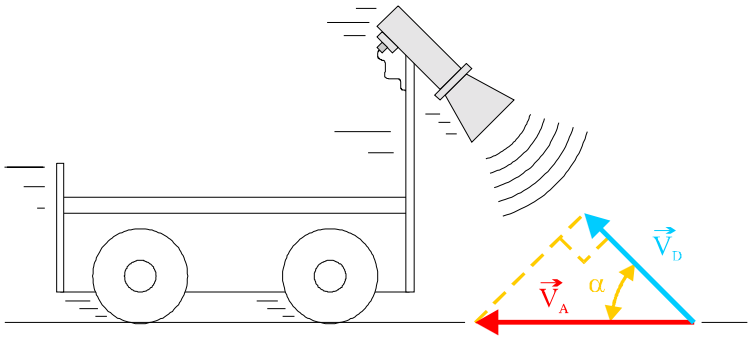


Figura 2.5: Medição de velocidade com sensor Doppler.

Everett (1995) e Borenstein *et al.* (1996) referem diversas fontes de erros na detecção da verdadeira velocidade do pavimento em relação a um veículo:

- Interferência dos lobos laterais.
- Componentes verticais da velocidade, introduzidas pela reacção de um veículo às anomalias da superfície do pavimento.
- Incertezas quanto ao verdadeiro ângulo de incidência, devidas à largura finita do feixe, que tornam necessário o recurso a técnicas de processamento de sinal.
- Ilusão provocada, por exemplo, por uma corrente de água que passa por cima do pavimento sobre o qual se encontra parado um veículo. Nessa situação, o sensor *Doppler* pode interpretar o movimento da água relativamente ao veículo como sendo o movimento do pavimento em relação ao veículo.

Na Tabela 2.6 apresentam-se algumas características do *Trak-Star*, sensor de velocidade por efeito *Doppler* fabricado pela Micro-Trak (Borenstein *et al.*, 1996).

Tabela 2.6: Características do *Trak-Star Ultrasonic Speed Sensor*.

Velocidade *(speed range)*	17,7m/s
Resolução de velocidade	1,8cm/s
Exactidão	±1,5%+0,04mph

A utilização de sensores Doppler não é uma solução habitual na navegação de robôs móveis, por ser mais dispendiosa e complexa que a odometria.

2.2.3 Utilização de Acelerómetros e Giroscópios

Na navegação inercial, que é independente de referências externas ao veículo, utilizam-se as chamadas *unidades de medição inercial*, constituídas por acelerómetros e giroscópios (Dougherty e Giardina, 1988; Soares e Restivo, 1997; Gu *et al.*, 2002).

"A saída de um acelerómetro é um sinal proporcional à variação de velocidade detectada ao longo do seu eixo de entrada" (Soares e Restivo, 1997). Para se obter a componente da posição do veículo segundo o eixo no qual o acelerómetro está montado, este sinal integra-se duas vezes (Dougherty e Giardina, 1988).

"A saída de um giroscópio é um sinal proporcional ao movimento angular em torno do seu eixo de entrada" (Soares e Restivo, 1997). Para se determinar quanto é que o veículo rodou em torno do eixo de entrada do giroscópio, este sinal integra-se uma vez.

Para determinar inercialmente a posição e a velocidade de um veículo que se pode mover livremente em qualquer direcção é necessário recorrer a unidades de medição inercial de três eixos, que possuem três giroscópios e três acelerómetros (Dougherty e Giardina, 1988; Soares e Restivo, 1997). Em muitas aplicações com robôs móveis o movimento faz-se a duas dimensões. Nesses casos basta recorrer a unidades de medição inercial de dois eixos, com dois acelerómetros e dois giroscópios.

Em testes realizados com acelerómetros aplicados à navegação de robôs móveis obteve-se geralmente um fraco desempenho pelas seguintes razões (Borenstein, 1994; Borenstein e Feng, 1994; Borenstein e Feng, 1996b; Borenstein *et al.*, 1997):

- Como os sinais provenientes dos acelerómetros têm de ser integrados duas vezes para se obter informação sobre a posição do veículo, assiste-se ao crescimento sem limites dos de erros de posição com o aumento da distância percorrida (Gu *et al.*, 2002; Venet *et al.*, 2002).

- Em condições típicas de funcionamento as acelerações podem ser muito pequenas, na ordem dos 0,01g. E são acelerações desta ordem de grandeza as que também ocorrem se o acelerómetro se desviar apenas 0,5º de uma posição perfeitamente horizontal, por exemplo quando se desloca sobre

superfícies irregulares. O resultado é que a relação sinal/ruído é muito pequena para acelerações baixas (curvas realizadas a baixa velocidade).

- Os acelerómetros são sensíveis a pavimentos irregulares, uma vez que qualquer perturbação relativamente a uma posição perfeitamente horizontal faz com que o sensor meça a aceleração da gravidade.

Os giroscópios podem ser mais exactos que os acelerómetros, mas são também mais dispendiosos e, além disso, sofrem do problema da deriva (que se mede em º/h): mesmo quando o veículo se encontra parado, o giroscópio continua a enviar um sinal ao sistema de navegação, indicando-lhe que o veículo se encontra em movimento (Borenstein, 1994; Borenstein e Feng, 1996b; Soares e Restivo, 1997; Di Marco *et al.*, 2000; Gu *et al.*, 2002). Isto é particularmente grave porque, como o sinal é integrado para se conhecer a orientação do veículo, verifica-se o crescimento sem limites de erros de orientação com o passar do tempo, o que também origina elevados erros de posição. Frappier *et al.* (1992), por exemplo, referem erros de posição de 10m ao fim de um percurso de 10km com uma unidade de medição inercial típica utilizável num veículo autónomo, em exteriores, num ambiente parcialmente estruturado.

Devido aos problemas expostos nos parágrafos anteriores, a navegação inercial de robôs móveis com acelerómetros e giroscópios não é geralmente considerada vantajosa em relação à odometria (Borenstein, 1994; Borenstein e Feng, 1996b). Como alternativa, Borenstein e Feng (1996a) propõem um método – a *girodometria* – que combina medidas provenientes de giroscópios com odometria. O principal ponto fraco dos sistemas de medição de posição baseados em odometria reside no facto de qualquer pequeno erro momentâneo de orientação produzir um erro de posição lateral que cresce constantemente. Com o auxílio de giroscópios torna-se possível detectar os erros de orientação no momento em que estes se produzem e proceder à sua imediata correcção.

Durante muito tempo, os giroscópios de grande exactidão (baixa deriva) eram demasiado dispendiosos para aplicações com robôs móveis. As unidades inerciais também são constituídas por acelerómetros, mas os giroscópios contribuem muito para o custo destes dispositivos (Soares e Restivo, 1997).

Os giroscópios de fibra óptica, muito exactos, já sofreram uma grande redução de preço e são actualmente uma solução muito atractiva para a navegação de robôs móveis.

Na Tabela 2.7 apresentam-se algumas características de giroscópios utilizados na navegação de robôs móveis (Borenstein *et al.,* 1996).

Tabela 2.7: Características de giroscópios utilizados na navegação de robôs móveis.

Características de Giroscópios		
Modelo	**Tipo**	**Deriva**
FP-G154 *(Futaba)*	*Rate Gyro* Mecânico	Dezenas de º/min
GyroEngine *(Gyration, Inc.)*	Mecânico	~9º/min
Gyrostar ENV-05 *(Murata)*	Piezoeléctrico	0,05º/s – 0,25º/s (3º/min – 15º/min)
-	*Closed-Loop* IFOG (Em desenvolvimento)	0,001º/h – 0,01º/h
Autogyro 3ARG-A (Saída analógica) 3ARG-D (Saída RS-232) *(Andrew Corp.)*	Fibra óptica	0,005º/s rms (18º/h rms)
Autogyro Navigator *(Andrew Corp.)*	Fibra óptica	0,005º/s rm (18º/h rms)
OFG-3 *(Hitachi Cable Ltd.)*	Fibra óptica	0,0028º/s (10º/h)

Em suma, os sistemas de navegação inercial possuem a vantagem de serem autónomos, não dependendo de referências externas. No entanto, a integração ao longo do tempo de pequenos erros constantes, inerente à utilização de giroscópios e acelerómetros, origina o crescimento sem limites dos erros de orientação e de posição com o tempo ou a distância percorrida. Por este motivo, estes sistemas de navegação não são adequados para efectuar medições de posição e orientação precisas durante longos períodos de tempo ou em trajectos longos.

2.3 Conclusões

Foram analisados diversos métodos que se podem utilizar na localização de robôs móveis. Os que permitem determinar a posição e/ou a orientação sem recorrer a suposições sobre movimentos anteriores destinam-se à localização absoluta. Os outros destinam-se à localização relativa. Todos se podem usar na autolocalização. A triangulação, a trilateração e o reconhecimento de marcos naturais também se usam na localização remota, mais difícil de implementar que a autolocalização quando há muitos veículos a localizar simultaneamente. A triangulação, a trilateração e o reconhecimento de marcos artificiais requerem que o ambiente seja preparado para efeitos de navegação.

É muito importante que um sistema de localização possua a capacidade de determinar a posição e a orientação de um robô móvel sem recorrer a suposições sobre movimentos anteriores uma vez que, em muitas situações, a informação sobre esses movimentos se pode perder ou não ser suficientemente fiável. De facto, os métodos destinados à localização relativa de robôs móveis são simples de usar, não dependem de referências externas e actualizam medidas a frequências elevadas, mas são geralmente pouco exactos para assegurar a navegação por períodos de tempo longos ou grandes distâncias (De Cecco, 2002).

A análise realizada permite concluir que também há métodos usados na localização absoluta que, em muitas circunstancias, são pouco exactos e/ou pouco fiáveis, nomeadamente

- a utilização de bússolas magnéticas, que são intoleravelmente inexactas em interiores devido a campos magnéticos locais;

- o reconhecimento de marcos, que só costuma produzir resultados suficientemente exactos quando o robô se encontra perto de um marco e a orientação do robô relativamente ao marco se mantém dentro de uma gama bem definida;

- o reconhecimento de marcos naturais e a correspondência de mapas que frequentemente produzem resultados pouco fiáveis;

- a trilateração com receptores GPS de baixo custo, que possuem uma exactidão de posicionamento da ordem de alguns metros, não funcionam em interiores e produzem medidas que são significativamente afectadas por interferências devidas à propagação multitrajectos.

Além disso, há métodos de localização absoluta que, em alguns casos, são demasiado lentos ou inadequados à localização contínua de robôs móveis. Isto aplica-se, por exemplo,

- ao reconhecimento de marcos, que é intermitente por natureza;

- ao reconhecimento de marcos naturais e à correspondência de mapas que, para serem suficientemente exactos, muitas vezes requerem um processamento demasiado demorado para a localização em tempo real[19];

- à trilateração com receptores GPS de baixo custo, que actualizam as medidas a uma frequência muito baixa.

Uma solução frequente é a de recorrer à fusão sensorial (Gu *et al.*, 2002) para combinar um método que faça continuamente a localização relativa com outro que, periodicamente, proceda à localização absoluta[20]. Pode também utilizar-se mais do que um método de cada tipo ou vários métodos do mesmo tipo. Eis algumas combinações usadas na autolocalização (Borenstein e Feng, 1996a; Borenstein *et al.*, 1996; Borenstein *et al.*, 1997; Soares e Restivo, 1997; Di Marco *et al.*, 2000; Wijk e Christensen, 2000; AGVE, 2001b; De Cecco, 2002; Clerentin *et al.*, 2002; EGEMIN, 2002c; AGVP, 2003a; Kleeman, 2003; Gning e Bonnifait, 2004):

- Utilização de unidades de medição inercial e trilateração com o *GPS;*

- Utilização de unidades de medição inercial e reconhecimento de marcos magnéticos;

[19] O reconhecimento de marcos e a correspondência de mapas baseados em visão por computador implicam uma complexidade e um custo que podem ser elevados (EGEMIN, 2002a) e que fazem com que a aplicabilidade destes métodos seja difícil de justificar em muitas situações.

[20] *"Mobile robot navigation can be considered as the art to overcome the inaccuracy of internal sensors and to take advantage of exteroceptive sensors like cameras, sonars or laser range finders to allow the robot move and act in its environment."* (Hayet *et al.*, 2002).

- Utilização de unidades de medição inercial, odometria e triangulação com balizas;

- Reconhecimento de marcos naturais e medição de distâncias com um telémetro laser;

- Odometria e reconhecimento de marcos;

- Odometria e utilização de um giroscópio;

- Odometria, utilização de um giroscópio e trilateração com o *GPS;*

- Odometria, utilização de balizas (com medição de ângulos e distâncias) e utilização de uma bússola magnética;

- Odometria e correspondência de mapas.

Os métodos de localização que não requerem a preparação do ambiente podem ser extremamente úteis – ou mesmo indispensáveis – à navegação em certos meios, sobretudo se estes forem desconhecidos ou não puderem ser modificados para efeitos de navegação. Mas quando a navegação se faz em ambientes conhecidos, bem estruturados e que podem ser modificados, é geralmente vantajoso recorrer a outros métodos, por exemplo à trilateração ou à triangulação com balizas. Estas técnicas podem ser implementadas em sistemas que recorrem à visão por computador (Yuen e MacDonald, 2002; Ji *et al.,* 2003). Existem, actualmente, sistemas de visão de baixo custo e dimensões reduzidas. No entanto, os sistemas que recorrem a sensores tais como *scanners laser*, apesar de maiores e mais dispendiosos, são muito mais exactos e possuem uma resolução muito maior (Hebert, 2000). Além disso, oferecem maior alcance, requerem muito menos processamento e actualizam as medidas a uma frequência muito superior.

A utilização de balizas possibilita a localização absoluta e contínua de um veículo ao longo de todos os trajectos que este possa percorrer mantendo uma comunicação em linha de vista com uma ou mais balizas. A localização faz-se habitualmente por triangulação, trilateração ou medição simultânea de ângulos e distâncias.

Nenhum dos sistemas de localização com balizas de radiofrequência funciona de maneira fiável em interiores. Isto aplica-se, em particular, ao GPS. Uma das principais limitações deste sistema é o facto de os sinais recebidos à superfície da Terra serem muito fracos, o que torna extremamente difícil a localização em interiores. Mesmo recorrendo a sofisticados receptores de alta sensibilidade, a exactidão de posicionamento que se pode obter é insuficiente para muitas aplicações. A localização com GPS em exteriores é intermitente na presença de alguns obstáculos e extremamente difícil em locais rodeados de edifícios altos. Mesmo em exteriores pouco obstruídos, o recurso exclusivo ao GPS não constitui uma solução adequada para medir em tempo real a posição de robôs móveis com uma exactidão da ordem dos milímetros:

- A exactidão de posicionamento é insuficiente;
- As interferências devidas à propagação multitrajectos dos sinais emitidos pelos satélites produzem frequentemente grandes erros de posicionamento;
- O tempo necessário para um receptor GPS determinar a sua posição pela primeira vez depois de ser ligado, ou depois de ter perdido o sincronismo com os satélites que estava a utilizar para se posicionar, pode variar entre 30s e 15m;
- O Departamento de Defesa dos Estados Unidos pode proceder à degradação intencional da qualidade dos sinais emitidos pelos satélites.

Os sistemas ópticos de localização com balizas são exactos, de custo relativamente baixo e funcionam de modo fiável em interiores. São geralmente mais exactos e mais económicos que os sistemas de radiofrequência. Os alcances da ordem das dezenas ou centenas de metros são suficientes em inúmeras aplicações. A velocidade de actualização das medidas é, em alguns casos, suficientemente elevada para dispensar o apoio de métodos de localização relativa. No entanto, a instalação e manutenção das balizas pode ser complexa e dispendiosa. Em NDC (1998) faz-se uma comparação entre sistemas que utilizam marcos magnéticos em combinação com sensores inerciais e sistemas ópticos com balizas baseados em *laser*. Apesar de a tecnologia ser significativamente mais dispendiosa, são bem patentes as vantagens dos

sistemas ópticos no que se refere à exactidão de navegação e à flexibilidade que oferecem.

Tendo em vista o objectivo definido no início deste capítulo é possível concluir que, entre os métodos analisados, a localização absoluta por trilateração ou triangulação com balizas é a mais adequada ao desenvolvimento de um sistema fiável e de custo relativamente baixo para a localização contínua em tempo real de robôs móveis que navegam com velocidades de alguns metros por segundo, em ambientes (exteriores ou interiores) quase-estruturados e não muito obstruídos. A tecnologia actualmente existente, nomeadamente a dos sistemas ópticos com balizas activas, permite obter uma exactidão de medição de posição da ordem dos milímetros e uma exactidão de medição de orientação da ordem das centésimas de grau, como pretendido.

No próximo capítulo investigar-se-á qual é, entre os diversos modos de implementar a trilateração e a triangulação com balizas, o melhor para a localização simultânea de vários robôs que navegam a duas dimensões.

3. Localização Absoluta com Balizas

Uma baliza é um dispositivo que, situado numa posição conhecida – mas não necessariamente fixa – do espaço, possibilita a localização absoluta e contínua de um veículo ao longo de todos os trajectos que este possa percorrer mantendo uma comunicação em linha de vista com uma ou mais balizas[1].

Na Figura 3.1 representa-se uma baliza que recorre à goniometria e à medição de distâncias para determinar a posição de um robô móvel[2]. Se essa informação for depois transmitida para o robô, este pode calcular a sua orientação se conhecer a posição da baliza e medir o ângulo λ_1. Este exemplo ilustra a possibilidade de a localização ser feita recorrendo a uma única baliza. No entanto, é muito frequente a utilização de diversas balizas integradas num sistema de localização.

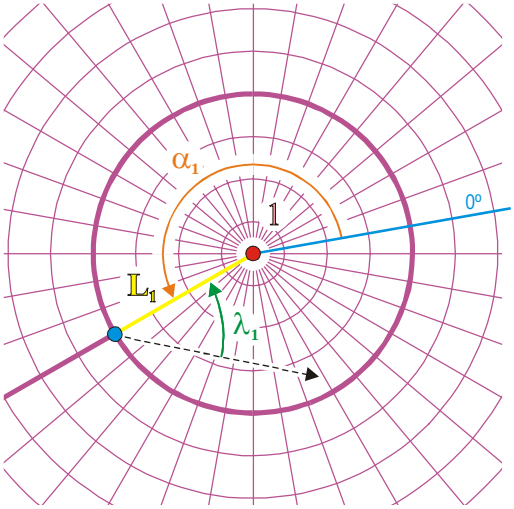


Figura 3.1: A baliza 1 determina a posição do robô medindo simultaneamente L_1 e α_1. Se o robô conhecer a própria posição e a posição da baliza pode determinar a sua orientação a partir da medição do ângulo λ_1.

[1] A partir deste capítulo, e salvo indicação contrária, nas figuras utilizar-se-ão círculos vermelhos para representar as balizas (pontuais) e um círculo azul para representar o robô móvel (pontual) que se pretende localizar. Uma seta a tracejado com origem no círculo azul indica um semieixo de referência fixo no robô. A orientação do robô é o ângulo que esse semieixo forma com o semieixo positivo dos xx do referencial ortonormado x0y definido no plano de navegação.

Não costuma ser uma aproximação válida considerar pontuais os robôs, sobretudo os que se movem em interiores. As dimensões desses robôs geralmente não são desprezáveis relativamente aos erros de posição admissíveis. A solução adoptada é considerar que a posição de um robô é a posição de um dos seus pontos.

[2] Trata-se do tipo de localização remota que caracteriza o *radar*.

Neste capítulo faz-se uma análise comparativa dos métodos de localização habitualmente utilizados em sistemas de localização com balizas. No ponto 3.1 descrevem-se algumas características gerais destes sistemas.

Nos pontos 3.2, 3.3 e 3.4 abordam-se, respectivamente,

- a localização baseada na medição de distâncias;
- a localização baseada na medição da diferença de distâncias;
- a localização baseada na goniometria.

No ponto 3.5 sugerem-se dois novos métodos de autolocalização:

- O primeiro baseia-se na medição simultânea de um ângulo orientado e de uma distância;
- O segundo baseia-se na medição simultânea de um ângulo orientado e da diferença de duas distâncias;

O capítulo termina com as conclusões e sugestões de trabalho futuro apresentadas no ponto 3.6.

3.1 Generalidades

As balizas utilizadas na localização remota são necessariamente activas, têm de ser ligadas em rede e, se houver vários veículos a localizar ao mesmo tempo, são mais complexas que as requeridas pela autolocalização porque têm de estar preparadas para receber simultaneamente sinais provenientes de vários veículos. Por isso, geralmente recorre-se à autolocalização quando há muitos veículos a localizar simultaneamente.

As diferentes balizas de um sistema de autolocalização são geralmente *distinguíveis*, ou seja, cada uma delas possui algum elemento identificador que a distingue de todas as outras. Um veículo tem de ser capaz de distinguir pelo menos uma baliza para se poder localizar utilizando a posição conhecida dessa baliza.

Um sistema de localização diz-se *activo* se entre o veículo e as balizas houver comunicação nos dois sentidos. Se houver comunicação em apenas um sentido (das

balizas para o veículo ou então do veículo para as balizas), o sistema de localização diz-se *passivo* (Everett, 1995; Borenstein *et. al.*, 1996; Drane e Rizos, 1998). Em geral verifica-se que:

- um *sistema activo de localização remota* requer o uso de balizas activas com a capacidade de emitir e receber;

- um *sistema passivo de localização remota* requer o uso de balizas activas receptoras;

- num *sistema activo de autolocalização* as balizas podem ser activas ou passivas mas, se houver mais do que um veículo a localizar-se ao mesmo tempo, têm de ser capazes de lidar simultaneamente com sinais provenientes de vários veículos;

- num *sistema passivo de autolocalização* a complexidade das balizas (activas ou passivas) não depende do número de veículos a localizar, uma vez que o sentido da comunicação é sempre de uma baliza para um veículo.

A localização pode ser feita a uma, duas ou três dimensões, dependendo do tipo de veículo e da aplicação concreta em causa. É indispensável localizar a três dimensões um avião. Mas um navio que atravessa o oceano pode ser localizado apenas a duas dimensões, uma vez que a terceira – a altitude – é conhecida (constante e igual a zero, neste caso). E para localizar um veículo que se desloca sobre carris ao longo de um corredor comprido, basta um sistema de localização a uma dimensão.

O objectivo deste trabalho é contribuir para o estudo de métodos de localização absoluta de robôs móveis que se movam num plano, pelo que apenas será analisada a localização a duas dimensões. Na ausência de outras restrições conhecidas à partida, um robô pode situar-se em qualquer ponto do plano de navegação. Cada medição realizada pelo sistema de localização define qual é, de acordo com essa medição, o lugar geométrico de todos os pontos nos quais o robô se pode encontrar[3]. Quando a localização se faz a duas dimensões, esse lugar geométrico é geralmente uma linha que

[3] O resultado da medição permanece constante desde que o robô se encontre em algum dos pontos desse lugar geométrico.

recebe o nome de *linha de posição*. A determinação da posição do robô faz-se mediante a intersecção de várias linhas de posição.

Um robô pode inclinar-se devido à irregularidade do pavimento ou às acelerações a que está sujeito quando se move. Isto pode dificultar ou até impedir a manutenção das comunicações em linha de vista com as balizas dos sistemas de autolocalização que utilizam sinais altamente direccionais. Pode também produzir erros de medição significativos, sobretudo quando se recorre à *autolocalização por goniometria* (ou *autolocalização por triangulação*).

A incerteza de medição faz com que exista uma incerteza associada à posição calculada, traduzida pelo facto de a intersecção das linhas de posição deixar de ser um ponto para se transformar numa superfície. As dimensões desta superfície dependem não só da incerteza de medição mas também das posições relativas do robô e das balizas. Em concreto, a incerteza de posição aumenta com:

a) o aumento da incerteza de medição;

b) a diminuição do menor dos ângulos formados por duas linhas de posição no ponto de cruzamento;

c) a diminuição da densidade de linhas de posição.

Para uma dada incerteza de medição, a situação mais favorável para minimizar a incerteza de posição é ter duas linhas de posição que se cruzam formando ângulos de 90º no ponto de intersecção, numa região com elevada densidade de linhas de posição.

3.2 Localização Baseada na Medição de Distâncias

A localização baseada na medição de distâncias é habitualmente designada *localização por trilateração.* A linha de posição resultante da medição da distância de um robô a uma baliza é uma circunferência centrada nessa baliza (Figura 3.2), independentemente de se recorrer à autolocalização ou à localização remota. A intersecção das duas circunferências obtidas com duas balizas permite determinar duas possíveis posições do robô (Figura 3.3). Esta ambiguidade pode ser ultrapassada recorrendo a uma terceira baliza que não seja colinear com as outras duas (Figura 3.4 e Figura 3.5).

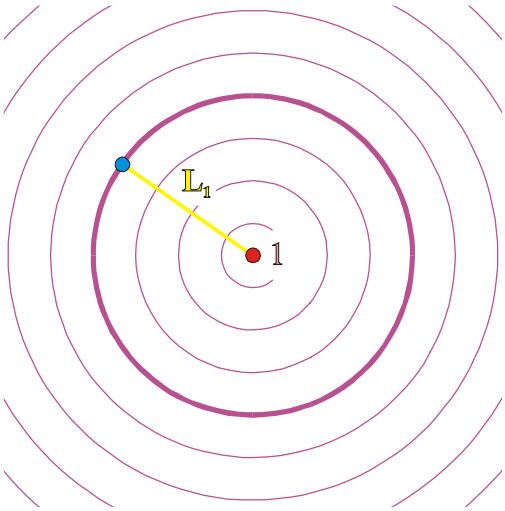


Figura 3.2: A linha de posição resultante da medição de L_1 é uma circunferência de raio L_1 centrada na baliza 1.

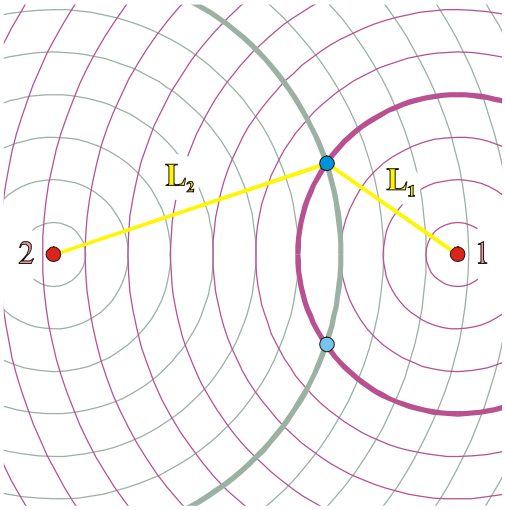


Figura 3.3: A intersecção das duas circunferências permite determinar duas possíveis posições do robô.

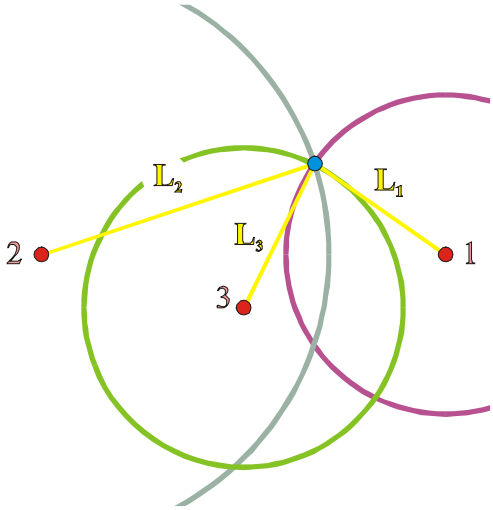


Figura 3.4: A utilização de uma terceira baliza não colinear com as outras duas permite determinar a posição do robô.

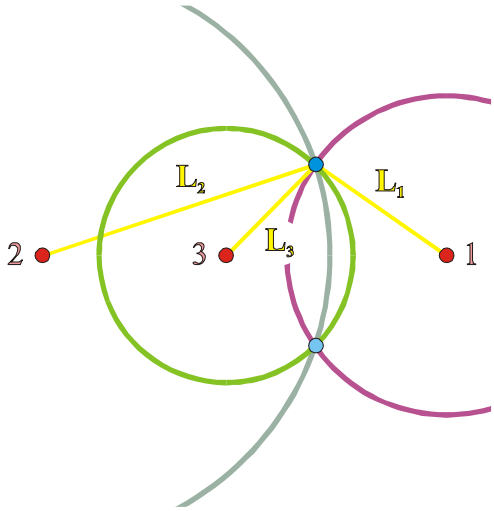


Figura 3.5: Utilizando três (ou mais) balizas colineares mantém-se a ambiguidade existente com apenas duas balizas.

McKerrow (1991), Everett (1995), Borenstein *et al.* (1996), Kelly (1996), Marques *et al.* (1996), Zhao (1997), Hebert (2000), Gu *et al.* (2002) e Kleeman (2003) descrevem diversos métodos de medir a distância a um alvo, aplicáveis à medição da distância entre um robô móvel e uma baliza (o alvo pode ser o robô ou uma baliza, dependendo do sistema). Os mais usados no âmbito da robótica recorrem a uma das seguintes técnicas:

1. Medição do tempo de propagação de sinais transmitidos entre o robô e cada baliza, partindo do princípio que os sinais se propagam em linha recta a uma velocidade constante e conhecida. Esta técnica requer que o sistema de localização seja activo (como acontece no *radar*) ou então que exista sincronização entre cada baliza e o robô (é o que ocorre no GPS). Tipicamente, conseguem-se alcances da ordem da centena de metros ou mais.

   Nos sistemas ópticos ou de radiofrequência os sinais propagam-se à velocidade da luz. Assim, a medição de distâncias a alvos muito próximos implica a medição de tempos muito pequenos, o que é dispendioso. Por este motivo, as distâncias mínimas que se podem medir com estes sistemas costumam ser elevadas[4]. Além disso, uma incerteza de medição de distâncias de 1mm requer uma incerteza de medição de tempos de apenas 3ps.

[4] Esta distância é de 1m para o *Laser Scanner 4-2.0* da *Danaher Motion* (DM, 2003) e de 1,2m para o *scanner* laser do sistema de navegação *NAV 200* da *SICK* (SICK, 2003).

A baixa velocidade de propagação dos ultra-sons no ar possibilita a medição de distâncias pequenas utilizando microcontroladores de baixo preço. Mas as distâncias máximas que se podem medir são geralmente inferiores a 20m. Além disso, a exactidão de medição é consideravelmente inferior à conseguida com sistemas baseados em *laser*. Um só sensor de ultra-sons típico não permite determinar o tamanho e a orientação dos alvos, o que facilita a ocorrência de erros de identificação.

2. Medição da diferença de fase existente entre o sinal emitido por uma fonte e a parte desse sinal que é reflectida por um alvo. Esta técnica requer que o sistema de localização seja activo. Permite obter uma frequência de actualização das medidas muito superior à que se consegue recorrendo à medição do tempo de propagação de sinais, mas a exactidão de medição decresce rapidamente com o aumento da distância medida. O alcance é geralmente limitado a algumas dezenas de metros, ou menos.

3. Triangulação, baseada no princípio da geometria plana segundo o qual um triângulo fica definido uma vez conhecidos o comprimento de um dos seus lados e os dois ângulos cujos vértices são as extremidades desse lado.

   Os *sistemas passivos de triangulação* utilizam apenas a luz ambiente que ilumina os alvos e recorrem habitualmente à análise de imagens. Se forem usadas câmaras de vídeo normais pode ser necessário iluminar artificialmente os alvos.

   Nos *sistemas activos de triangulação* os alvos são iluminados de uma forma específica por uma fonte de energia controlada pelo próprio sistema, o que dispensa a utilização de luz ambiente especial. Por isso, são muito menos sensíveis às condições ambientais. Pode não ser viável a utilização de alguns destes sistemas na localização simultânea de vários robôs.

   A triangulação pode ser implementada de várias maneiras:

   - Os *sistemas passivos de visão estereoscópica* utilizados na medição de distâncias recorrem a duas câmaras separadas por uma distância conhecida, dispostas de forma a ver simultaneamente o mesmo alvo. A imagem observada por uma das câmaras encontra-se deslocada

relativamente à imagem observada pela outra câmara[5]. Este deslocamento chama-se *disparidade* e é inversamente proporcional à distância ao alvo. Nestes sistemas, que têm experimentado importantes avanços[6], a principal dificuldade é estabelecer uma correspondência entre os pontos observados por uma das câmaras e os pontos observados pela outra.

- Nos *sistemas activos de visão estereoscópica* o problema da correspondência de pontos das imagens observadas por cada câmara continua a existir, mas pode ser consideravelmente simplificado.

- Há sistemas activos de triangulação nos quais uma das câmaras é substituída por uma fonte de luz apontada ao alvo, eliminando o problema da correspondência de pontos de imagens inerente aos sistemas de visão estereoscópica.

  A fonte pode produzir um simples *feixe luminoso* que é reflectido pelo alvo e produz um ponto na imagem captada pela câmara restante, numa posição que depende da distância ao alvo. Em vez da câmara, que é um sensor bidimensional, pode-se utilizar um sensor unidimensional sujeito a um movimento de rotação. A distância ao alvo é calculada a partir da medida da orientação do sensor no momento em que este detecta o feixe luminoso reflectido. Desta forma, evita-se a análise de imagens.

  Também se pode recorrer a uma fonte de *luz estruturada* que, em vez de um simples feixe luminoso, produz um padrão de luz (uma linha, por exemplo) que é projectado sobre o alvo. A câmara permite observar as distorções sofridas pelo padrão projectado ao atingir o alvo. É com base nestas distorções que se calcula a distância ao alvo.

- É possível medir a distância a um alvo utilizando simples trigonometria, com um sistema passivo de triangulação e recorrendo a apenas uma

[5] As duas imagens de um sistema de visão estereoscópica podem ser captadas por uma única câmara sucessivamente colocada nos dois pontos de observação, desde que o alvo permaneça em repouso relativamente a esses dois pontos. Se o alvo estiver em movimento relativamente aos pontos de observação, esta técnica não é adequada à localização absoluta no sentido previamente definido, uma vez que é necessário ter em conta o movimento do alvo no período de tempo decorrido entre a captação das duas imagens.

[6] Os sistemas de medição de tempo de propagação de sinais baseados em *laser* são muito mais exactos, possuem uma resolução muito superior à dos sistemas passivos de visão estereoscópica e oferecem maior alcance. Mas ainda são demasiado grandes e dispendiosos para muitas aplicações. Nos últimos anos registaram-se notáveis avanços ao nível da algoritmia, da capacidade de computação e da redução de tamanho dos sistemas computacionais e sensoriais. Como resultado, já existem sistemas passivos de visão estereoscópica de reduzido tamanho e baixo custo. São uma solução eficaz para muitos problemas de robótica que não requerem os elevados níveis de exactidão e resolução oferecidos pelos sistemas baseados em *laser* e que não podiam ser resolvidos com esses sistemas (Hebert, 2000).

câmara, tendo em conta que a imagem do alvo aumenta à medida que este se aproxima da câmara. Mas a correcta aplicação do método requer que as dimensões do alvo sejam conhecidas e que esse alvo se encontre numa direcção perpendicular ao eixo óptico da câmara[7]. Há também sistemas activos de triangulação que, recorrendo ao mesmo princípio, utilizam um feixe de raios laser controlado como fonte luminosa e um simples sensor em vez da câmara.

- Duas imagens de um robô em movimento, captadas em instantes sucessivos a partir de um único ponto de uma baliza fixa, podem ser usadas para calcular por triangulação a distância da baliza ao robô, desde que seja conhecido o trajecto percorrido pelo robô entre a captação de cada imagem. Alternativamente, podem usar-se duas imagens de uma baliza fixa captadas em instantes sucessivos a partir de um único ponto de um robô em movimento. Assim, os dois pontos de observação necessários ao cálculo da distância podem obter-se com um único sensor (por exemplo, uma câmara). No entanto, estas técnicas não são de localização absoluta (tal como esta se definiu previamente) porque, para calcular a distância entre o robô e uma baliza num dado instante, é necessário conhecer movimentos anteriores do robô.

Todos os métodos de medição de distâncias por triangulação sofrem do problema das *partes em falta (missing parts),* que consiste no facto de haver pontos aos quais não se pode medir a distância porque esses pontos:

a) não são simultaneamente visíveis dos dois pontos de observação, nos sistemas de visão estereoscópica;

b) não são iluminados pela fonte luminosa ou não reflectem a luz recebida de forma a que esta incida no ponto de observação, nos sistemas activos com apenas uma câmara ou sensor unidimensional rotativo;

[7] Pode ser muito difícil garantir esta condição, a não ser em casos particulares. Alguns sistemas resolvem este problema utilizando alvos cilíndricos.

c) estão sobre um alvo que não é integralmente visto porque está muito próximo do ponto de observação e/ou possui dimensões demasiado grandes, nos sistemas que recorrem a alvos de dimensões conhecidas.

O problema das partes em falta agrava-se quando se aumenta a exactidão de medição de distâncias recorrendo a um dos seguintes processos:

a) aumento da distância que existe entre os dois pontos de observação, nos sistemas de visão estereoscópica;

b) aumento da distância que existe entre a fonte e o receptor de luz, nos sistemas activos;

c) aumento das dimensões dos alvos, nos sistemas que recorrem a alvos de dimensões conhecidas.

O alcance dos sistemas de medição de distâncias por triangulação depende muito da exactidão de medição requerida. Os sistemas activos usados na robótica móvel exibem, tipicamente, alcances de alguns centímetros a alguns metros. Nos sistemas passivos, o alcance pode chegar a ser da ordem da centena de metros.

Na autolocalização de robôs de pequenas dimensões, a distância máxima que é possível estabelecer entre os dois pontos de observação dos sistemas de visão estereoscópica ou entre a fonte e o receptor de luz dos sistemas activos pode ser de tal modo pequena que só é possível garantir uma elevada exactidão de medição de distâncias se o alcance for muito reduzido.

Independentemente do método utilizado para medir distâncias, a medição da orientação absoluta – ou seja, sem recorrer a suposições sobre movimentos anteriores – do robô requer:

- a determinação simultânea das posições de pelo menos dois pontos do robô[8], na localização remota;

- medições feitas simultaneamente a partir de dois pontos do robô[8], na autolocalização.

O valor de orientação obtido é tanto mais correcto quanto mais afastados um do outro estiverem os dois pontos do robô utilizados em cada caso. Se o robô for de dimensões reduzidas pode não ser possível conseguir um afastamento que garanta a exactidão de medição necessária[9].

3.3 Localização Baseada na Medição da Diferença de Distâncias

Também se costuma chamar *localização por trilateração* à localização baseada na medição da diferença de distâncias. O tempo de propagação de um sinal transmitido entre o robô e uma baliza é proporcional à distância que os separa. Se duas balizas emitirem sinais idênticos em fase, o robô recebe dois sinais que só estão em fase se as balizas estiverem à mesma distância do robô. Sendo conhecida a velocidade de propagação dos sinais, o desfasamento entre os dois sinais recebidos permite determinar a diferença de distâncias entre o robô e cada baliza. Esta técnica, que pode ser implementada com sistemas passivos de localização, requer apenas sincronização entre as balizas (Zhao, 1997; Drane e Rizos, 1998), que é geralmente mais fácil de conseguir do que a sincronização entre cada baliza e o robô. O princípio também é válido se for o robô a emitir um sinal que é recebido por duas balizas.

À semelhança do que ocorre com a localização baseada na medição de distâncias o cálculo da orientação absoluta requer a determinação simultânea das posições de pelo menos dois pontos do robô ou então medições feitas simultaneamente a partir de dois pontos do robô, dependendo do tipo de localização.

A linha de posição resultante da medição da diferença das distâncias de um robô a duas balizas é uma hipérbole cujos focos se situam nessas balizas (Figura 3.6),

[8] É possível calcular a orientação do robô determinando, em instantes sucessivos, a posição de um único ponto do robô ou então recorrendo a medições feitas em instantes sucessivos a partir de um único ponto do robô. No entanto, nenhuma destas técnicas está de acordo com a definição de orientação absoluta previamente adoptada.

[9] Para se conseguir uma exactidão de 0,08º com o receptor GPS *Trimble MS860*, as duas antenas deste receptor devem estar separadas de 5m. A separação deve ser de 10m para se obter uma exactidão de 0,03º (Trimble, 2000).

independentemente de se recorrer à autolocalização ou à localização remota. A intersecção de duas hipérboles obtidas com três ou mais balizas permite determinar a posição do robô (Figura 3.7), desde que essa intersecção seja um só ponto.

Duas hipérboles podem intersectar-se em mais do que um ponto (Figura 3.8), dando origem a uma ambiguidade na posição calculada. Essa ambiguidade pode ser ultrapassada recorrendo a mais balizas. A utilização de mais do que três balizas também pode ser feita para se obter uma geometria favorável. A configuração de balizas representada na Figura 3.9, por exemplo, proporciona uma vasta zona com elevada densidade de linhas de posição que se intersectam formando ângulos próximos de 90º nos pontos de intersecção (Kelly, 1996).

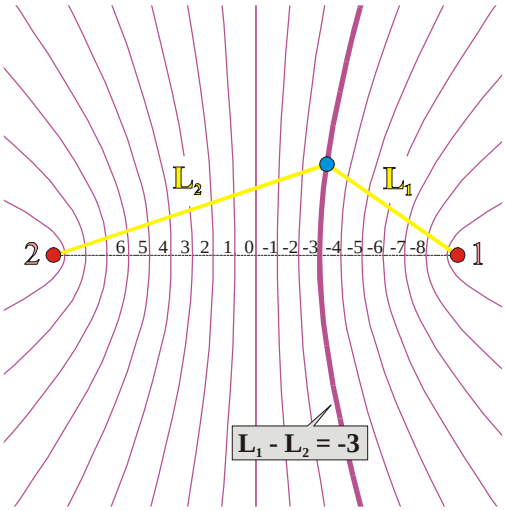


Figura 3.6: A linha de posição resultante da medição de $L_1 - L_2$ é uma hipérbole cujos focos se situam nas balizas 1 e 2.

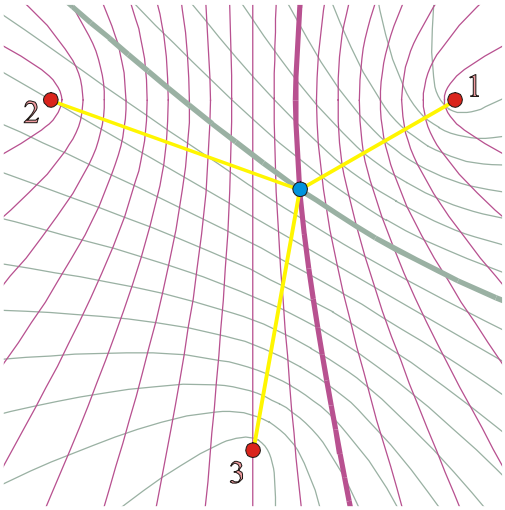


Figura 3.7: A intersecção das duas hipérboles permite determinar a posição do robô.

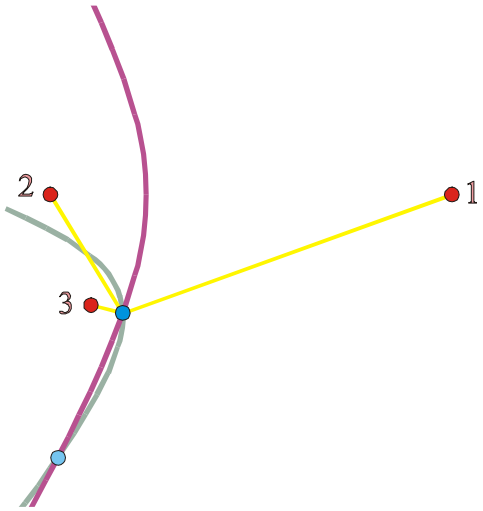


Figura 3.8: Duas hipérboles podem intersectar-se em mais do que um ponto, dando origem a uma ambiguidade na posição calculada.

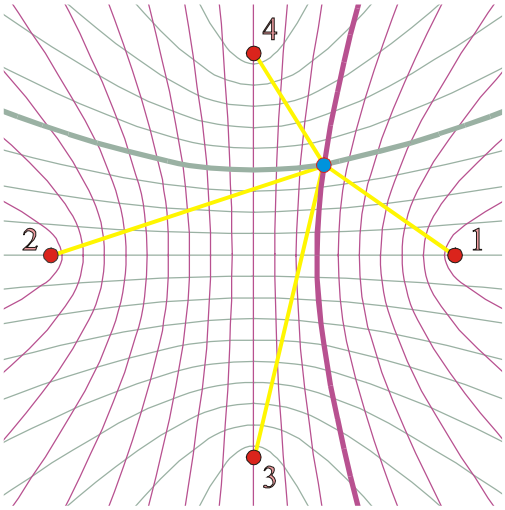


Figura 3.9: Configuração de balizas que proporciona uma vasta zona com elevada densidade de linhas de posição que se intersectam formando ângulos próximos de 90º nos pontos de intersecção.

3.4 Localização Baseada na Goniometria

A localização baseada na goniometria, habitualmente designada *localização por triangulação*, pode ser implementada com sistemas passivos de localização e não requer sincronização entre cada baliza e o robô. A medição de ângulos é mais simples que a medição de distâncias, podendo fazer-se recorrendo simplesmente a um sensor rotativo associado a um *encoder*, por exemplo. Ao contrário do que ocorre no cálculo de distâncias por medição do tempo de propagação de sinais, não é necessário garantir uma distância mínima entre o robô e cada baliza, mesmo quando se recorre a sistemas ópticos. E, como a medição de um ângulo se pode fazer a partir de um só ponto, não

ocorre o problema das partes em falta que caracteriza a medição de distâncias por triangulação. O estudo realizado no Capítulo 2 revela que existe actualmente a tecnologia necessária para se obter uma exactidão de medição de ângulos da ordem das centésimas de grau em sistemas de localização com alcances de centenas de metros[10]. No entanto, os erros de posição e de orientação produzidos por interferências devidas à propagação multitrajectos são maiores na localização baseada na goniometria do que a localização baseada na medição de distâncias ou de diferenças de distâncias, sobretudo se houver oclusão de sinais[11] (Zhao, 1997).

3.4.1 Localização Remota

Para se efectuar a localização remota por triangulação recorre-se a um mínimo de duas balizas dotadas de um goniómetro. Cada baliza mede um ângulo orientado tal que o lado origem é um semieixo de referência e o lado extremidade é o segmento de recta que une a baliza ao robô (Figura 3.10). A linha de posição resultante é uma semi-recta com origem na baliza. A intersecção das duas semi-rectas obtidas com as duas balizas permite determinar a posição do robô (Figura 3.11).

Se o robô se situar sobre a recta que une as balizas não é possível determinar univocamente a sua posição. Esta dificuldade pode ser ultrapassada recorrendo a uma terceira baliza não colinear com as outras duas (Figura 3.12).

À semelhança dos métodos anteriormente descritos, o cálculo da orientação absoluta do robô requer a determinação simultânea das posições de pelo menos dois dos seus pontos.

A principal vantagem da localização remota por triangulação reside no facto de dispensar o uso de goniómetros a bordo do robô. No entanto, possui as desvantagens inerentes à operação dos sistemas remotos de localização quando há muitos robôs a localizar simultaneamente.

[10] O *CONAC (Computerized Opto-electronic Navigation and Control)* da *MTI Research, Inc)* é um sistema passivo de autolocalização por triangulação cujas balizas são emissores laser rotativos. Pelas suas excelentes características, pode considerar-se uma referência importante (Everett, 1995; Borenstein et. al., 1996). As exactidões de medição de posição e de orientação são, respectivamente, ±1,3mm e ±0,05º, as medidas são actualizadas à frequência de 20Hz e o alcance é de 250m.

[11] Por este motivo, é mais aconselhavel recorrer à trilateração quando as comunicações entre um robô e cada baliza se fazem por radiofrequência.

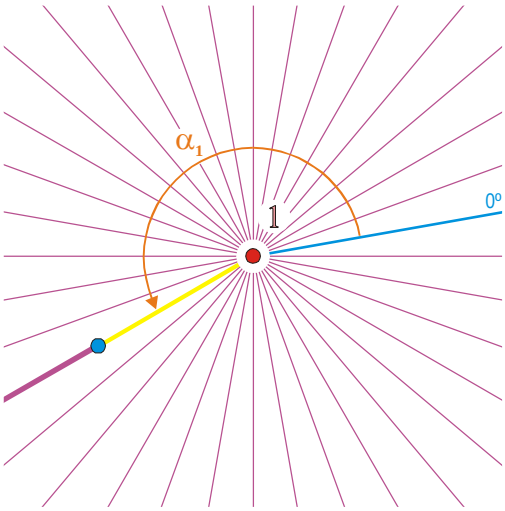


Figura 3.10: A linha de posição resultante da medição do ângulo orientado α_1 é uma semi-recta com origem na baliza 1.

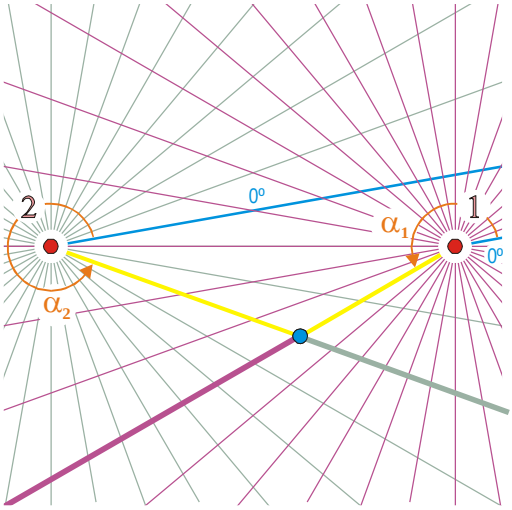


Figura 3.11: A intersecção das duas semi-rectas permite determinar a posição do robô.

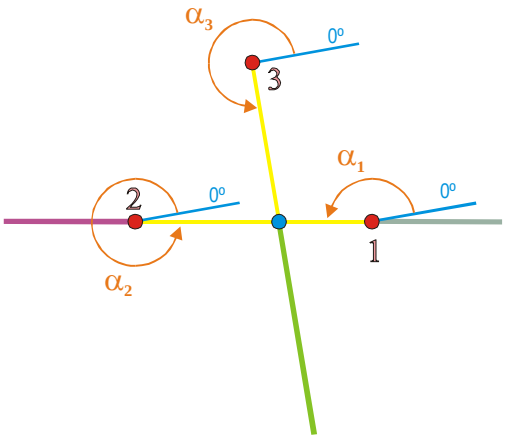


Figura 3.12: A utilização da baliza 3, não colinear com as balizas 1 e 2, permite que seja determinada a posição do robô quando este se encontra sobre a recta definida pelas balizas 1 e 2.

3.4.2 Autolocalização

Um robô pode determinar a sua própria posição recorrendo apenas a duas balizas (Anexo A) se a sua orientação puder ser determinada sem se recorrer à triangulação (com uma bússola, por exemplo). Seja λ_1 (Figura 3.13) um ângulo orientado tal que o seu lado origem é um semieixo de referência fixo no robô e o seu lado extremidade é o segmento de recta que une o robô à baliza 1. A linha de posição que resulta da medição de λ_1 é uma semi-recta com origem na baliza 1. Utilizando outra baliza obtém-se uma segunda semi-recta que se intersecta com a primeira no ponto em que se encontra o robô (Figura 3.14).

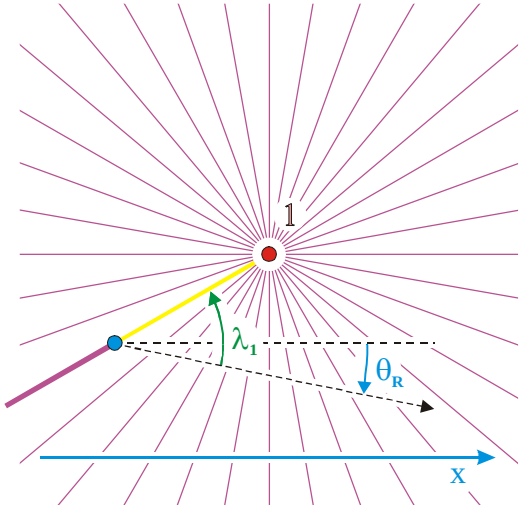


Figura 3.13: Com θ_R conhecido, a linha de posição resultante da medição do ângulo orientado λ_1 é uma semi-recta com origem na baliza 1

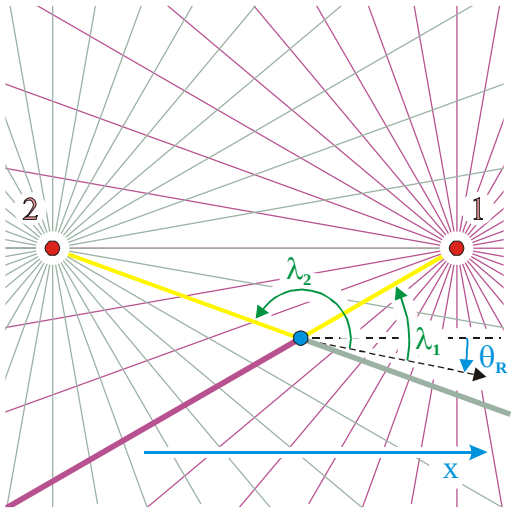


Figura 3.14: A intersecção de duas semi-rectas permite determinar a posição do robô.

À semelhança do que acontece na localização remota por triangulação, se o robô se situar sobre a recta que une as balizas não é possível determinar univocamente a sua posição. Esta dificuldade pode ser ultrapassada recorrendo a uma terceira baliza não colinear com as outras duas (Figura 3.15).

Duas balizas são insuficientes para a autolocalização no caso de a orientação do robô ser desconhecida. Seja λ_{12} (Figura 3.16) um ângulo orientado tal que $0^o \leq \lambda_{12} < 360^o$. O seu lado origem é o segmento de recta que une o robô à baliza 1 e o seu lado extremidade é o segmento de recta que une o robô à baliza 2. Mostrar-se-á no Capítulo 5 que a linha de posição que resulta da medição de λ_{12} é um arco de circunferência cujas extremidades são as balizas 1 e 2.

Recorrendo a uma terceira baliza obtém-se outro arco de circunferência (Figura 3.17) que, em geral, intersecta o primeiro apenas no ponto em que se situa a baliza comum aos dois arcos e no ponto em que se encontra o robô[12]. Desta forma é geralmente possível determinar sem ambiguidade a posição do robô recorrendo a apenas três balizas, ou seja, resolver o Problema de Pothenot. No entanto, quando o robô se encontra sobre a circunferência definida por três balizas não colineares, a intersecção dos dois arcos é também um arco e não um ponto. Algo semelhante ocorre quando o robô se encontra sobre a recta definida por três balizas colineares: as linhas de posição obtidas com cada par de balizas são semi-rectas ou segmentos de recta (arcos de raio infinito) cuja intersecção é um segmento de recta ou uma semi-recta e não um ponto. Nestas situações – que serão estudadas com mais detalhe nos próximos capítulos – o robô não se consegue localizar.

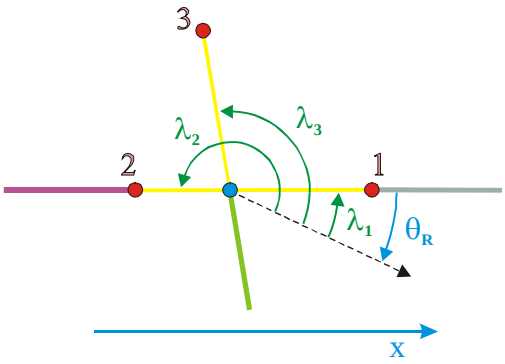


Figura 3.15: A utilização da baliza 3, não colinear com as balizas 1 e 2, permite que o robô determine a sua posição quando se encontra sobre a recta definida pelas balizas 1 e 2.

[12] Dois arcos de circunferência não sobrepostos e com um ponto comum só podem intersectar-se em mais um ponto.

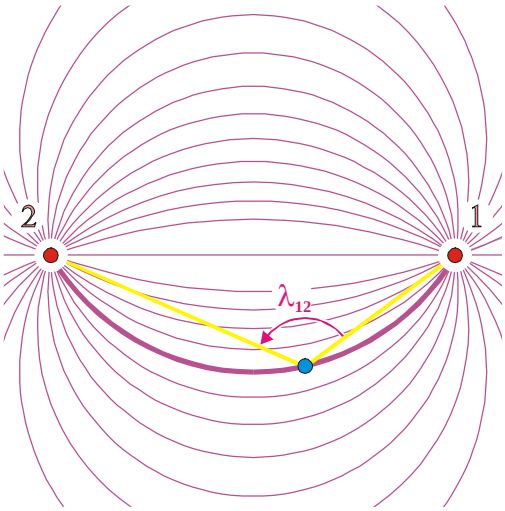


Figura 3.16: A linha de posição resultante da medição do ângulo orientado λ_{12} é um arco de circunferência compreendido entre as balizas 1 e 2.

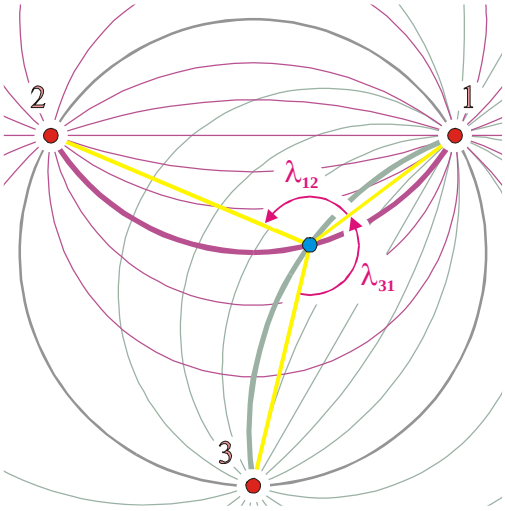


Figura 3.17: A intersecção dos dois arcos permite determinar a posição do robô.

Já foi referido que a localização por triangulação dispensa a sincronização entre as balizas e o robô. No caso particular da autolocalização, também não é necessária a sincronização entre balizas. Além disso, ao contrário do que ocorre na localização remota por triangulação e na trilateração, o robô pode determinar a sua orientação recorrendo apenas às medições efectuadas num mesmo instante e a partir de um só ponto (uma vez calculada a posição do robô, a sua orientação pode determinar-se a partir da medição do ângulo orientado formado por um semieixo de referência fixo no

robô com o segmento de recta que une o robô a uma das balizas). A posição do robô também pode ser calculada recorrendo exclusivamente a essas medições.

A autolocalização a duas dimensões por triangulação tem, no entanto, a desvantagem de não ser adequada se houver inclinações pronunciadas do robô, nomeadamente as que podem ocorrer na navegação sobre superfícies irregulares ou com desníveis. Nessas situações é melhor recorrer à localização remota por triangulação ou à trilateração.

3.5 Localização Baseada na Medição Simultânea de Duas Grandezas

Neste ponto sugerem-se dois novos métodos de autolocalização baseados na medição simultânea de duas grandezas e que recorrem apenas a duas balizas:

- O primeiro método requer que, a partir de um robô que se pretende autolocalizar, se meçam simultaneamente um ângulo orientado formado pelos segmentos que unem o robô a duas balizas[13] e a distância a uma delas, como se mostra na Figura 3.18. O robô encontra-se na intersecção de dois arcos de circunferência. Este método possui a complexidade inerente à medição simultânea de ângulos e distâncias.

- O segundo método requer que, a partir de um robô que se pretende autolocalizar, se meçam simultaneamente um ângulo orientado formado pelos segmentos que unem o robô a duas balizas[13] e a diferença das distâncias do robô a cada uma dessas balizas (Figura 3.19). O robô encontra-se na intersecção de um arco de circunferência com uma hipérbole. As linhas de posição cruzam-se formando ângulos próximos de 90º numa extensa zona entre as balizas. Este método possui a complexidade inerente à medição simultânea de ângulos e diferenças de distâncias.

Em ambos os métodos, uma vez calculada a posição do robô, a sua orientação pode determinar-se a partir da medição do ângulo orientado formado por um semieixo de referência fixo no robô com o segmento de recta que une o robô a uma das balizas (o processo é o mesmo utilizado na autolocalização por triangulação).

[13] Como já foi referido – e será demonstrado no Capítulo 5 – a linha de posição que resulta da medição deste ângulo orientado é um arco de circunferência cujas extremidades são as duas balizas.

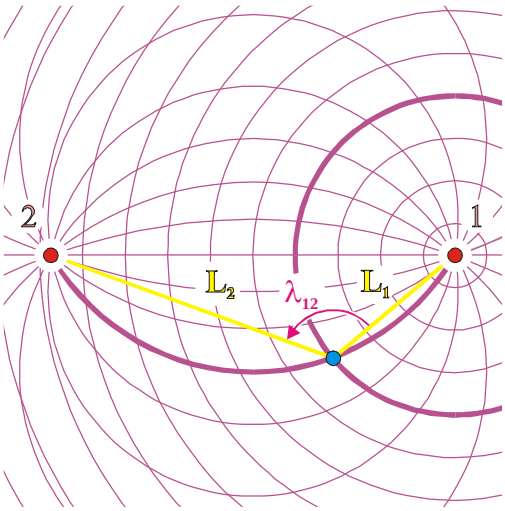


Figura 3.18: O robô determina a sua posição medindo simultaneamente um ângulo orientado formado pelos segmentos que o unem a duas balizas e a distância a uma delas.

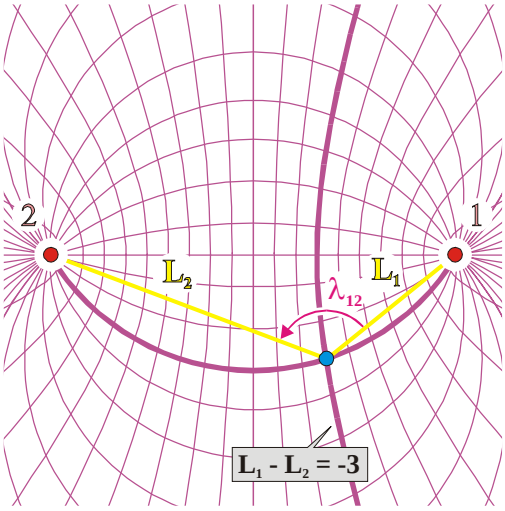


Figura 3.19: O robô determina a sua posição medindo simultaneamente um ângulo orientado formado pelos segmentos que o unem a duas balizas e a diferença das distâncias a cada uma dessas balizas.

3.6 Conclusões

A autolocalização com sistemas passivos é a mais adequada quando há muitos robôs a localizar simultaneamente.

Na localização por medição de distâncias, habitualmente designada *localização por trilateração,* as técnicas mais utilizadas na robótica requerem uma das seguintes modalidades:

- Implementação com sistemas activos;
- Sincronização entre cada baliza e o robô;
- Medições feitas a partir de dois pontos tão afastados um do outro quanto possível;
- Utilização de alvos de dimensões conhecidas.

Sobre a localização baseada na medição de distâncias é ainda possível tecer as seguintes considerações:

- Quando a medição de distâncias se baseia no tempo de propagação de sinais transmitidos entre o robô e as balizas, as distâncias mínimas que se podem medir com os sistemas mais exactos costumam ser elevadas.
- Quando a medição de distâncias se baseia na medição da diferença de fase de sinais, a exactidão de medição decresce rapidamente com o aumento da distância medida. O alcance típico encontra-se limitado a algumas dezenas de metros, ou menos.
- Muitos sistemas de medição de distâncias por triangulação são complexos e todos os métodos de medição de distâncias por triangulação sofrem do problema das *partes em falta*. O alcance destes sistemas depende muito da exactidão de medição requerida. Na autolocalização de robôs de pequenas dimensões, para garantir uma elevada exactidão de medição, o alcance pode ter de ser muito reduzido.
- É possível medir a distância a um alvo utilizando simples trigonometria, com um sistema passivo de triangulação e recorrendo a apenas um sensor, tendo em conta que a imagem do alvo aumenta à medida que este se aproxima da câmara. Mas a correcta aplicação deste método requer que as dimensões do alvo sejam conhecidas e que esse alvo se encontre numa direcção perpendicular ao eixo óptico do sensor.

- Independentemente do método utilizado para medir distâncias, a medição da orientação absoluta – ou seja, sem recorrer a suposições sobre movimentos anteriores – do robô requer:

  - a determinação simultânea das posições de pelo menos dois pontos do robô, na localização remota;

  - medições feitas simultaneamente a partir de dois pontos do robô, na autolocalização.

  O valor de orientação obtido é tanto mais correcto quanto mais afastados um do outro estiverem os dois pontos do robô necessários em cada caso. Se o robô for de reduzidas dimensões pode não ser possível conseguir um afastamento que garanta a exactidão de medição necessária. As alternativas são as seguintes:

  - Determinação, em instantes sucessivos, da posição de um único ponto do robô, na localização remota;

  - Medições feitas em instantes sucessivos a partir de um único ponto do robô, na autolocalização.

  Nenhum destes métodos é de orientação absoluta, tal como esta se definiu no Capítulo 2.

Também se chama *localização por trilateração* à localização baseada na medição de diferença de distâncias, que pode ser implementada com sistemas passivos de localização e requer apenas sincronização entre as balizas. Esta é geralmente mais fácil de conseguir do que a sincronização entre cada baliza e o robô. Tal como acontece na localização baseada na medição de distâncias, o cálculo da orientação absoluta requer a determinação simultânea das posições de pelo menos dois pontos do robô ou então medições feitas simultaneamente a partir de dois pontos do robô, dependendo do tipo de localização.

A localização baseada na goniometria, habitualmente designada *localização por triangulação*, pode ser implementada com sistemas passivos de localização e não requer

sincronização entre cada baliza e o robô. A medição de ângulos é mais simples que a medição de distâncias, sendo verdade que:

- ao contrário do que ocorre no cálculo de distâncias por medição do tempo de voo de sinais, não é necessário garantir uma distância mínima entre o robô e cada baliza, mesmo quando se recorre a sistemas ópticos.

- como a medição de um ângulo se pode fazer a partir de um único ponto, não ocorre o problema das partes em falta que caracteriza a medição de distâncias por triangulação.

- as interferências devidas à propagação multitrajectos afectam mais a localização baseada na goniometria do que a localização baseada na medição de distâncias ou de diferenças de distâncias, sobretudo se houver oclusão de sinais.

- a localização remota por triangulação dispensa o uso de goniómetros a bordo do robô. No entanto, possui as desvantagens inerentes à operação dos sistemas remotos de localização quando há muitos robôs a localizar simultaneamente. Além disso, à semelhança do que ocorre com a trilateração, o cálculo da orientação absoluta do robô requer a determinação simultânea das posições de pelo menos dois dos seus pontos.

- um robô que navega num plano pode determinar a sua própria posição recorrendo a apenas duas balizas se a sua orientação puder ser determinada sem se recorrer à triangulação.

- na localização remota por triangulação e na autolocalização por triangulação com orientação conhecida, se o robô se situar sobre a recta que une as duas balizas requeridas não é possível determinar univocamente a sua posição. Esta dificuldade pode ser ultrapassada recorrendo a uma terceira baliza não colinear com as outras duas.

- a autolocalização por triangulação dispensa não só a sincronização entre as balizas e o robô, mas também a sincronização entre balizas. Além disso, ao contrário do que ocorre na localização remota por triangulação e na

trilateração, o robô pode determinar a sua orientação recorrendo apenas às medições efectuadas num mesmo instante e a partir de um único ponto. A posição do robô também pode ser calculada recorrendo exclusivamente a essas medições.

- a autolocalização a duas dimensões por triangulação pode ser feita sem ambiguidade recorrendo a apenas três balizas, excepto quando o robô se encontra sobre a circunferência definida por três balizas não colineares ou a recta definida por três balizas colineares.

- a autolocalização a duas dimensões por triangulação não é adequada se houver inclinações pronunciadas do robô, nomeadamente as que podem ocorrer na navegação sobre superfícies irregulares ou com desníveis. Nessas situações, é mais adequado recorrer à localização remota por triangulação ou à trilateração.

De acordo com o exposto, entre todos os métodos de trilateração e de triangulação analisados, a autolocalização absoluta por triangulação – baseada na resolução do Problema de Pothenot – é a solução mais adequada[14] para a localização simultânea de vários robôs que navegam a duas dimensões, desde que não ocorram inclinações significativas desses robôs. Este método pode ser implementado com sistemas passivos de localização, não requer a sincronização entre as balizas e o robô nem a sincronização entre balizas e é o único que permite que o robô determine a sua orientação recorrendo exclusivamente às medições efectuadas num mesmo instante e a partir de um só ponto. A posição do robô também pode ser calculada recorrendo exclusivamente a essas medições. No próximo capítulo analisar-se-ão diversos algoritmos de triangulação.

No ponto 3.5 apresentaram-se dois novos métodos que permitem a autolocalização recorrendo apenas a duas balizas, e cujo estudo aprofundado se sugere como trabalho futuro:

- O primeiro método requer que, a partir de um robô que se pretende autolocalizar, se meçam simultaneamente um ângulo orientado formado pelos

[14] Quando apareceu pela primeira vez, o já referido *CONAC* era um sistema de localização remota. Posteriormente, surgiu numa nova versão: um sistema passivo de autolocalização por triangulação (Everett, 1995; Borenstein et. al., 1996).

segmentos que unem o robô a duas balizas e a distância a uma delas. Possui a complexidade inerente à medição simultânea de ângulos e distâncias.

- O segundo método requer que, a partir de um robô que se pretende autolocalizar, se meçam simultaneamente um ângulo orientado formado pelos segmentos que unem o robô a duas balizas e a diferença das distâncias do robô a cada uma dessas balizas. Possui a complexidade inerente à medição simultânea de ângulos e diferenças de distâncias.

Em ambos os métodos, uma vez calculada a posição do robô, a sua orientação pode determinar-se a partir da medição do ângulo orientado formado por um semieixo de referência fixo no robô com o segmento de recta que une o robô a uma das balizas.

4. Algoritmos de Triangulação com Três Balizas

O estudo realizado no capítulo anterior tornou patente as vantagens da autolocalização absoluta de robôs móveis por triangulação com balizas. Neste capítulo faz-se a análise comparativa de diversos algoritmos de triangulação destinados à autolocalização absoluta, a duas dimensões, com sistemas passivos de localização.

No ponto 4.1 define-se o problema da autolocalização absoluta por triangulação.

No ponto 4.2 aborda-se a ambiguidade de posição que consiste no facto de, a um dado conjunto de medidas de ângulos, corresponder mais do que uma posição possível no plano de navegação.

No ponto 4.3 analisam-se as restrições que são comuns a todos os algoritmos de autolocalização por triangulação.

Apesar de haver algoritmos que utilizam mais de três balizas (Betke e Gurvits, 1997, Ji *et al.*, 2003), esse número é geralmente suficiente para a autolocalização. Neste capítulo serão apenas analisados, nos pontos 4.4 a 4.12, algoritmos de triangulação com três balizas[1]. Os que se descrevem a partir do ponto 4.5 têm sido utilizados ou referidos no âmbito da robótica móvel.

No ponto 4.13 apresentam-se as conclusões deste capítulo.

Parte-se do princípio que todas as balizas referidas neste capítulo são emissoras omnidireccionais pontuais e distinguíveis, com padrão de emissão isotrópico e alcance infinito. Assume-se também que todos os ângulos medidos pelo robô o são a partir de um mesmo ponto do plano de navegação[2] e que o robô não muda a sua orientação enquanto as medições estão a ser feitas. Estas duas condições ficam garantidas se os

[1] Cohen e Koss (1992) definem os algoritmos que recorrem apenas a três balizas como sendo de *triangulação com três objectos*.

[2] Em rigor, os métodos que utilizam medidas obtidas em pontos diferentes não são de localização absoluta (no sentido previamente definido) uma vez que envolvem cálculos relativos a movimentos do robô anteriores ao instante em que se dispõe de todos os dados que permitem determinar a sua localização.

ângulos forem todos medidos num mesmo instante. Isto não é sempre possível, nomeadamente se os ângulos forem medidos por um sensor rotativo[3].

4.1 Definição do Problema da Autolocalização Absoluta por Triangulação

O problema da autolocalização absoluta a duas dimensões por triangulação pode colocar-se nos seguintes termos: *dadas três balizas distinguíveis situadas em posições conhecidas de um plano de navegação no qual se definiu um referencial ortonormado x0y e os valores assumidos, num dado instante, por dois dos três ângulos[4] existentes entre os segmentos de recta que unem o robô a cada baliza e por um ângulo existente entre um semieixo de referência fixo no robô e o segmento de recta que une o robô a uma das balizas, determinar sem recorrer a suposições sobre movimentos anteriores:*

- *a posição do robô, ou seja, as suas coordenadas x_R e y_R no referencial x0y;*
- *a orientação do robô, ou seja, o ângulo θ_R que o semieixo de referência fixo no robô forma com o semieixo positivo dos xx do referencial x0y.*

4.2 Ambiguidade de Posição

Em geral, quando a navegação se faz num plano, tem de haver pelo menos três balizas visíveis para que um robô se possa autolocalizar por triangulação. Seja λ_{12} o menor dos ângulos formados pelos segmentos de recta que unem um robô a duas balizas (1 e 2) não colineares com o robô[5] (Figura 4.1). A medida deste ângulo determina dois arcos de circunferência – cujas extremidades são as balizas 1 e 2 – sobre os quais o robô se pode encontrar, no plano de navegação[6] (Kuipers e Levitt, 1988; Sutherland e Thompson, 1993; Sutherland, 1994; Araújo, 1999). Recorrendo a uma terceira baliza (3)

[3] Nesse caso, é necessário garantir que a velocidade de rotação do sensor é suficientemente elevada face à velocidade de actualização de localização pretendida. Esta depende da velocidade do robô que se está a localizar.

[4] Estes ângulos podem ser calculados a partir dos ângulos existentes entre um semieixo de referência fixo no robô e os segmentos de recta que unem o robô a cada baliza.

[5] Isto implica $\lambda_{12} < 180º$.

[6] Se o robô e as duas balizas forem colineares:

- se o robô se encontrar entre as duas balizas, então $\lambda_{12} = 180º$ e os dois arcos degeneram no segmento de recta que une as duas balizas, que pode ser visto como um único arco de circunferência de raio infinito.
- se o robô não se encontrar entre as duas balizas, então $\lambda_{12} = 0º$ e os dois arcos degeneram em duas semi-rectas com origem nas balizas, direcção do segmento de recta, que une as balizas e não contêm esse segmento. As semi-rectas podem ser vistas como um único arco de circunferência de raio infinito.

pode medir-se λ_{31}, menor dos ângulos formados pelos segmentos de recta que unem o robô às balizas 1 e 3. Este ângulo determina mais dois arcos que se intersectam com os primeiros na baliza 1 e no ponto em que se encontra o robô. No entanto, os quatro arcos podem intersectar-se em outros pontos (Figura 4.1 e Figura 4.2), dando origem a uma ambiguidade na determinação da posição do robô.

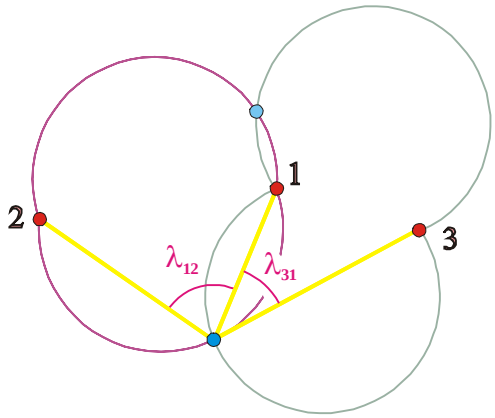


Figura 4.1: Nesta situação, há duas posições a partir das quais se medem os ângulos λ_{12} e λ_{31} representados.

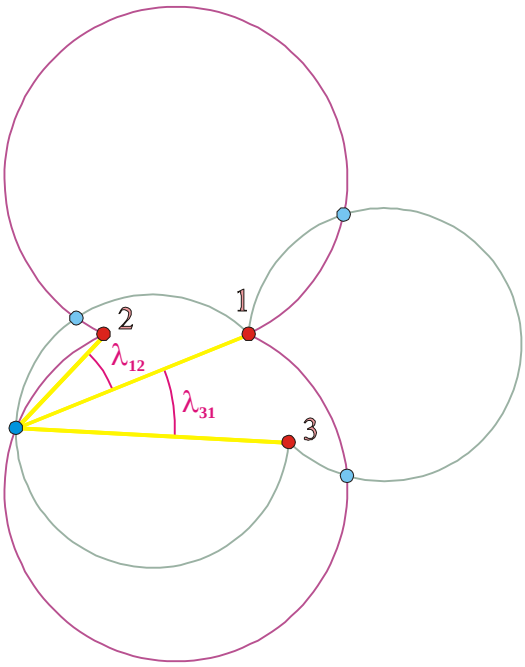


Figura 4.2: Nesta situação, há quatro posições a partir das quais se medem os ângulos λ_{12} e λ_{31} representados.

Na prática, esta ambiguidade pode ser ultrapassada de várias maneiras. Dois exemplos:

- Na situação da Figura 4.3 um robô está constrangido a navegar dentro de um recinto de navegação delimitado por uma fronteira conhecida. Apesar de haver sempre duas posições a partir das quais se podem medir cada par de ângulos λ_{12} e λ_{31}, só uma dessas posições se encontra dentro do recinto de navegação, pelo que a outra se pode excluir.

- Quando se recorre ao auxílio de um método de localização relativa (odometria, por exemplo), o robô geralmente conhece de antemão qual é a sua posição aproximada. As intersecções de arcos que ocorram muito longe desta posição podem ser excluídas.

No Capítulo 5 mostrar-se-á um modo de reduzir a ambiguidade na determinação da posição do robô, recorrendo a ângulos orientados.

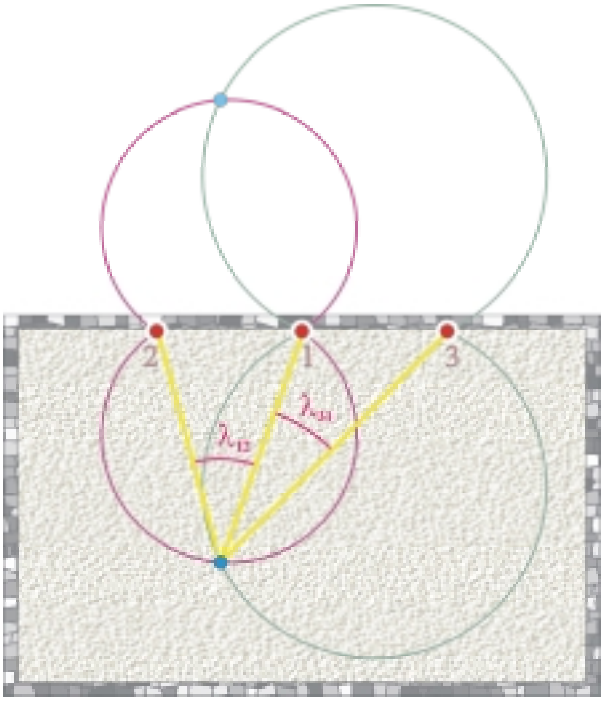


Figura 4.3: Nesta situação, há duas posições a partir das quais se medem os ângulos λ_{12} e λ_{31} representados. No entanto, só uma dessas posições se encontra dentro do recinto de navegação, pelo que a outra se pode excluir.

4.3 Restrições Comuns a Todos os Algoritmos de Autolocalização por Triangulação

A autolocalização por triangulação não é possível nos pontos do plano de navegação a partir dos quais se avistam menos de três balizas. Nesses pontos é necessário fazer a autolocalização por outro método. Mesmo que o plano de navegação esteja livre de obstáculos, surge uma dificuldade se o robô passar por um ponto no qual uma baliza é ocultada por outra baliza. Esta dificuldade pode resolver-se mudando a altura a que se encontram as balizas – por exemplo, colocando-as no tecto – de tal forma que todas elas sejam sempre visíveis. Mas então surge uma segunda dificuldade: o instrumento de medição de ângulos deve ter a capacidade de reconhecer simultaneamente duas balizas que possuam a mesma orientação relativamente ao eixo de referência fixo no robô. Se o instrumento não possuir essa capacidade ou se cada baliza puder ser ocultada por qualquer uma das outras duas, então a autolocalização por triangulação com três balizas não é possível numa das seguintes situações:

- Quando o robô se encontra sobre a recta definida por três balizas colineares;
- Quando o robô se encontra sobre alguma das semi-rectas representadas a cheio na (Figura 4.4), se as três balizas forem não colineares.

Três balizas simultaneamente visíveis são habitualmente suficientes para a autolocalização por triangulação. No entanto, isto não é verdade se o robô se encontrar sobre a circunferência definida por três balizas não colineares. A autolocalização não é possível nesse caso porque a intersecção dos arcos correspondentes às medidas de λ_{12} e λ_{31} não é um ponto, nem sequer um conjunto finito de pontos, mas sim um arco dessa circunferência, como se pode ver na (Figura 4.5).

A circunferência definida por três balizas não colineares degenera numa recta quando estas mudam de posição e ficam colineares. Ainda que as três balizas permaneçam sempre visíveis, as medidas de λ_{12} e λ_{31} realizadas quando o robô se encontra sobre essa recta só podem ser 0º ou 180º e apenas permitem determinar qual é a parte da recta (um segmento de recta ou uma semi-recta) sobre a qual se encontra o robô.

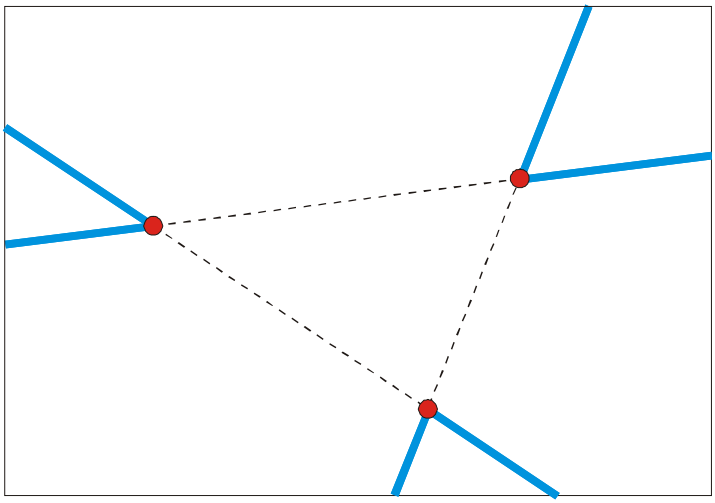


Figura 4.4: Se o instrumento de medição de ângulos não tiver a capacidade de reconhecer simultaneamente duas balizas que possuam a mesma orientação relativamente ao eixo de referência fixo no robô ou se cada baliza puder ser ocultada por qualquer uma das outras duas, então a autolocalização por triangulação com três balizas não é possível quando o robô se encontra sobre alguma das semi-rectas representadas a cheio.

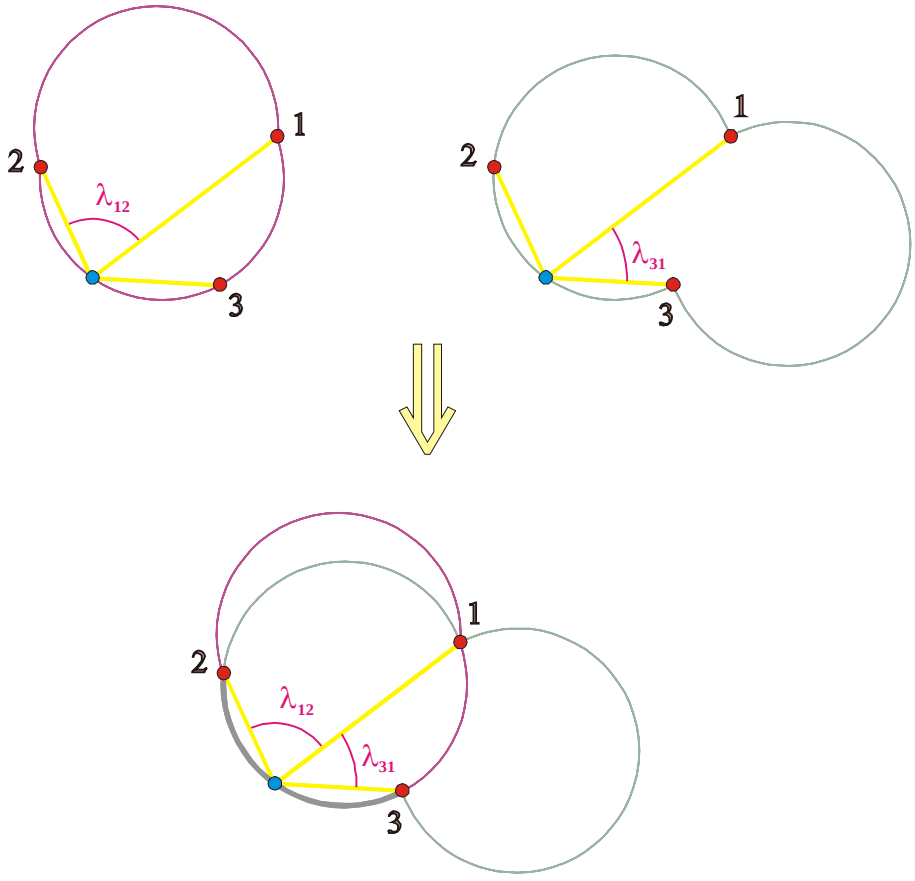


Figura 4.5: A autolocalização não é possível se o robô se encontrar sobre a circunferência definida por três balizas não colineares porque a intersecção dos arcos correspondentes às medições de λ_{12} e λ_{31} é um arco dessa circunferência e não um ponto (ou um conjunto finito de pontos).

Assim, independentemente do algoritmo de triangulação utilizado, mesmo que todas as balizas sejam visíveis de qualquer ponto e o instrumento de medição de ângulos possua a capacidade de reconhecer simultaneamente várias balizas que tenham a mesma orientação relativamente ao eixo de referência fixo no robô, há duas situações nas quais a autolocalização por triangulação com três balizas nunca é possível:

- Quando o robô se encontra sobre a circunferência definida por três balizas não colineares (Figura 4.6);

- Quando o robô se encontra sobre a recta definida por três balizas colineares (Figura 4.7).

Seguidamente, estudar-se-ão diversos algoritmos de triangulação com três balizas. Verificar-se-á se cada algoritmo, além de estar sujeito às duas restrições apontadas, também possui limitações específicas. Não se pretende analisar exaustivamente todas as limitações de cada algoritmo, mas apenas detectar as mais importantes. Admitir-se-á, em cada caso, que não ocorrem erros de medição de ângulos.

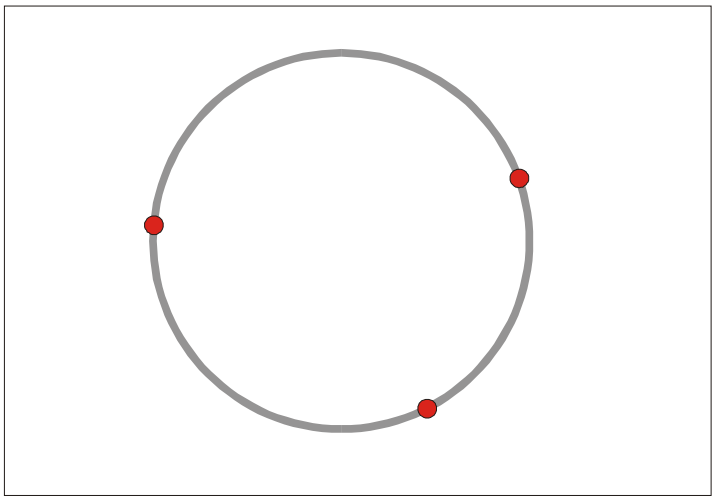


Figura 4.6: A autolocalização por triangulação não é possível quando o robô se encontra sobre a circunferência definida por três balizas não colineares.

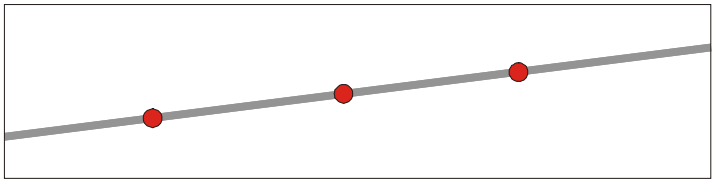


Figura 4.7: A autolocalização por triangulação não é possível quando o robô se encontra sobre a recta definida por três balizas colineares.

4.4 Algoritmo Simples de Triangulação

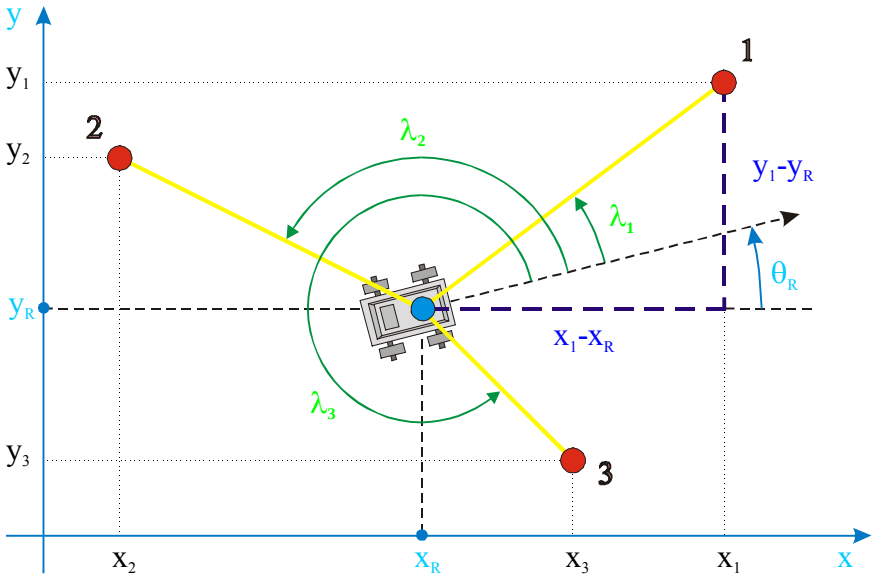
Um algoritmo simples de triangulação, que poderia ser usado na determinação de $\mathbf{x_R}$, $\mathbf{y_R}$ e $\boldsymbol{\theta_R}$ (Figura 4.8), consiste na resolução de um sistema de três equações não lineares:

1. Para determinar $\mathbf{x_R}$, $\mathbf{y_R}$ e $\boldsymbol{\theta_R}$ (Figura 4.8) resolver o seguinte sistema de equações não lineares em $\mathbf{x}$, $\mathbf{y}$ e $\boldsymbol{\theta}$:

$$\begin{cases} \dfrac{y_1 - y}{x_1 - x} = \operatorname{tg}(\lambda_1 + \theta) \\ \dfrac{y_2 - y}{x_2 - x} = \operatorname{tg}(\lambda_2 + \theta) \\ \dfrac{y_3 - y}{x_3 - x} = \operatorname{tg}(\lambda_3 + \theta) \end{cases}$$

Há duas soluções matematicamente possíveis para a orientação do robô: θ_R e θ_R+180º (Anexo B). Por isso, o robô precisa de conhecer à partida um valor aproximado da sua orientação.

Nos pontos que se seguem apresentam-se algoritmos mais complexos, mas que envolvem cálculos mais fáceis de executar.



x_1 , y_1 - Coordenadas da baliza **1**
x_2 , y_2 - Coordenadas da baliza **2**
x_3 , y_3 - Coordenadas da baliza **3**
x_R , y_R - Coordenadas do robô no referencial x0y
θ_R - Orientação do robô

λ_1 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **1**
λ_2 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **2**
λ_3 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **3**

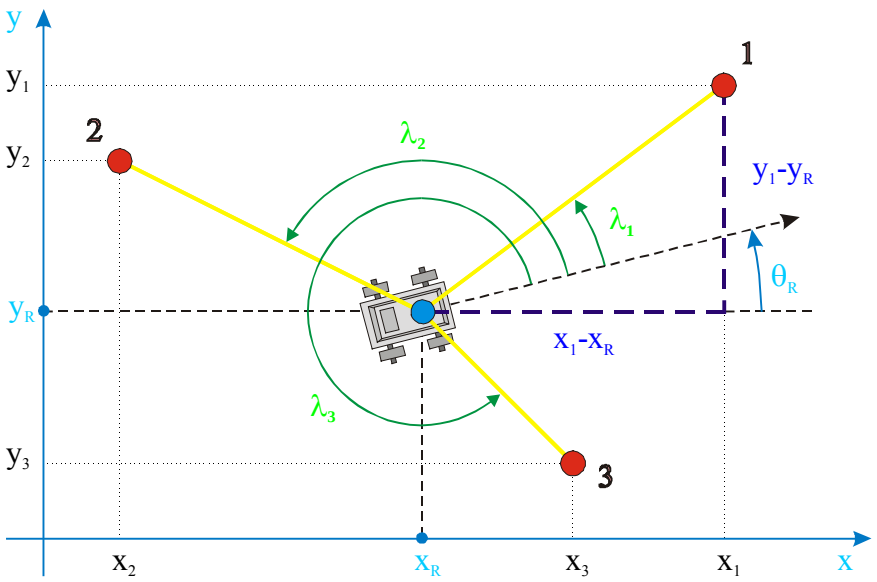
Figura 4.8: Grandezas em jogo no Algoritmo Simples de Triangulação.

4.5 Algoritmo de Triangulação Baseado na Pesquisa Iterativa

O Algoritmo de Triangulação Baseado na Pesquisa Iterativa é formulado por Cohen e Koss (1992) do seguinte modo:

1. Para uma orientação entre –90º e 90º com incrementos de 0,1º fazer;
   a) calcular a posição do robô a partir das balizas 1 e 2, por triangulação com essas balizas;
   b) calcular a posição do robô a partir das balizas 1 e 3, por triangulação com essas balizas;
   c) calcular a posição do robô a partir das balizas 2 e 3, por triangulação com essas balizas;
   d) se um erro foi enviado por as balizas e o robô serem colineares, então a triangulação não pode ser feita. Enviar um erro;
   e) calcular o perímetro do triângulo formado pelos pontos encontrados nas alíneas a), b) e c);
   f) guardar as coordenadas de posição correspondentes ao menor perímetro até agora determinado;
2. Enviar a solução.

A orientação calculada pertence sempre ao intervalo [–90º, +90º], podendo ser necessário adicionar 180º ao resultado obtido. Para isso, o robô precisa de conhecer à partida um valor aproximado da sua orientação verdadeira θ_R (Figura 4.9).



x_1 , y_1 - Coordenadas da baliza **1**

x_2 , y_2 - Coordenadas da baliza **2**

x_3 , y_3 - Coordenadas da baliza **3**

$\Delta\theta$ - Resolução angular

λ_1 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **1**

λ_2 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **2**

λ_3 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **3**

x_{RA} ,y_{RA} - Estimativa da posição do robô feita, em cada iteração, a partir das balizas **1** e **2,** usando o valor de a **θ** correspondente a essa iteração.

x_{RB} ,y_{RB} - Estimativa da posição do robô feita, em cada iteração, a partir das balizas **2** e **3,** usando o valor de a **θ** correspondente a essa iteração.

x_{RC} ,y_{RC} - Estimativa da posição do robô feita, em cada iteração, a partir das balizas **1** e **3,** usando o valor de a **θ** correspondente a essa iteração.

P_Δ - Perímetro do triângulo formado, em cada iteração, pelas estimativas de posição (x_{RA} ,y_{RA}), (x_{RB} ,y_{RB}) e (x_{RC} ,y_{RC}).

x_R ,y_R - Posição do robô.

θ_R - Orientação do robô.

Figura 4.9: Grandezas em jogo no Algoritmo de Triangulação Baseado na Pesquisa Iterativa.

A formulação de Cohen e Koss (1992) admite a seguinte especificação (Anexo C), que considera as grandezas indicadas na Figura 4.9:

1. Para $\theta = -90^\circ$ fazer:

$$\begin{cases} x_{RA} = \dfrac{y_1 - y_2 - x_1 \cdot tg(\lambda_1 + \theta) + x_2 \cdot tg(\lambda_2 + \theta)}{tg(\lambda_2 + \theta) - tg(\lambda_1 + \theta)} \\ y_{RA} = y_1 - (x_1 - x_{RA}) \cdot tg(\lambda_1 + \theta) \end{cases}$$

$$\begin{cases} x_{RB} = \dfrac{y_2 - y_3 - x_2 \cdot tg(\lambda_2 + \theta) + x_3 \cdot tg(\lambda_3 + \theta)}{tg(\lambda_3 + \theta) - tg(\lambda_2 + \theta)} \\ y_{RB} = y_2 - (x_2 - x_{RB}) \cdot tg(\lambda_2 + \theta) \end{cases}$$

$$\begin{cases} x_{RC} = \dfrac{y_1 - y_3 - x_1 \cdot tg(\lambda_1 + \theta) + x_3 \cdot tg(\lambda_3 + \theta)}{tg(\lambda_3 + \theta) - tg(\lambda_1 + \theta)} \\ y_{RC} = y_1 - (x_1 - x_{RC}) \cdot tg(\lambda_1 + \theta) \end{cases}$$

$$\begin{cases} P_\Delta \equiv \sqrt{(x_{RA} - x_{RB})^2 + (y_{RA} - y_{RB})^2} + \sqrt{(x_{RB} - x_{RC})^2 + (y_{RB} - y_{RC})^2} + \sqrt{(x_{RA} - x_{RC})^2 + (y_{RA} - y_{RC})^2} \\ \theta_R \equiv \theta \\ x_R \equiv x_{RA} \\ y_R \equiv y_{RA} \end{cases}$$

2. Para θ a variar de $-90^\circ + \Delta\theta$ até $+90^\circ$ em incrementos de $\Delta\theta$ fazer:

$$\begin{cases} x_{RA} = \dfrac{y_1 - y_2 - x_1 \cdot tg(\lambda_1 + \theta) + x_2 \cdot tg(\lambda_2 + \theta)}{tg(\lambda_2 + \theta) - tg(\lambda_1 + \theta)} \\ y_{RA} = y_1 - (x_1 - x_{RA}) \cdot tg(\lambda_1 + \theta) \end{cases}$$

$$\begin{cases} x_{RB} = \dfrac{y_2 - y_3 - x_2 \cdot tg(\lambda_2 + \theta) + x_3 \cdot tg(\lambda_3 + \theta)}{tg(\lambda_3 + \theta) - tg(\lambda_2 + \theta)} \\ y_{RB} = y_2 - (x_2 - x_{RB}) \cdot tg(\lambda_2 + \theta) \end{cases}$$

$$\begin{cases} x_{RC} = \dfrac{y_1 - y_3 - x_1 \cdot tg(\lambda_1 + \theta) + x_3 \cdot tg(\lambda_3 + \theta)}{tg(\lambda_3 + \theta) - tg(\lambda_1 + \theta)} \\ y_{RC} = y_1 - (x_1 - x_{RC}) \cdot tg(\lambda_1 + \theta) \end{cases}$$

Se

$$\sqrt{(x_{RA} - x_{RB})^2 + (y_{RA} - y_{RB})^2} + \sqrt{(x_{RB} - x_{RC})^2 + (y_{RB} - y_{RC})^2} + \sqrt{(x_{RA} - x_{RC})^2 + (y_{RA} - y_{RC})^2} \le P_\Delta$$

então

$$\begin{cases} P_\Delta \equiv \sqrt{(x_{RA} - x_{RB})^2 + (y_{RA} - y_{RB})^2} + \sqrt{(x_{RB} - x_{RC})^2 + (y_{RB} - y_{RC})^2} + \sqrt{(x_{RA} - x_{RC})^2 + (y_{RA} - y_{RC})^2} \\ \theta_R \equiv \theta \\ x_R \equiv x_{RA} \\ y_R \equiv y_{RA} \end{cases}$$

O algoritmo não funciona quando o robô e duas balizas são colineares[7]. Além disso, o processo de pesquisa faz com que seja centenas de vezes mais lento que, por exemplo, os algoritmos apresentados nos pontos 4.6, 4.7 e 4.10 (Cohen e Koss 1992).

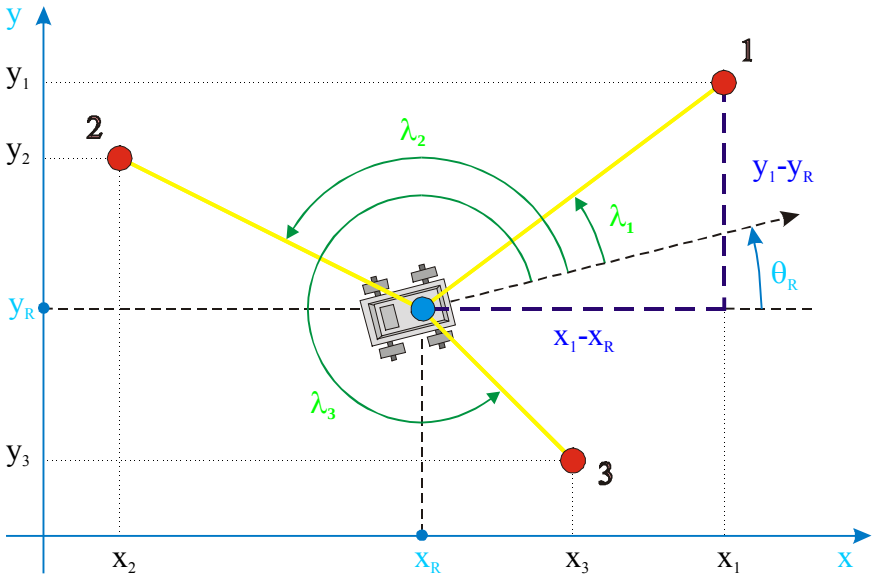
[7] Há também indeterminações do tipo $\frac{\infty}{\infty}$ quando algum dos ângulos $\lambda_i+\theta$ (i=1, 2 ou 3) se torna igual a 90º ou 270º. Para levantar estas indeterminações torna-se necessário recorrer à manipulação das expressões apresentadas.

4.6 Algoritmo de Triangulação Baseado no Método de Newton-Raphson

O Algoritmo de Triangulação baseado no Método de Newton-Raphson (Cohen e Koss, 1992) foi desenvolvido por Yuval Roth. A sua formulação geral é a seguinte:

1. Determinar a posição estimada do robô;
2. Dispor os dados em matrizes apropriadas à decomposição triangular LU;
3. Para o número desejado de iterações, fazer o seguinte:
   a) Resolver equações lineares recorrendo à decomposição triangular LU;
   b) Verificar se a raiz convergiu e, se sim, então enviar uma resposta;
   c) Actualizar a solução.

Esta formulação pode ser ampliada de forma a incluir o cálculo da orientação e especificada como se segue, considerando as grandezas indicadas na Figura 4.10.



x_1 , y_1 - Coordenadas da baliza **1**

x_2 , y_2 - Coordenadas da baliza **2**

x_3 , y_3 - Coordenadas da baliza **3**

n - número máximo de iterações a executar

ε'_x - erro relativo tolerado em **x**

ε'_y - erro relativo tolerado em **y**

ε'_θ - erro relativo tolerado em **θ**

ε - erro tolerado em $\mathbf{f}_1$, $\mathbf{f}_2$ e $\mathbf{f}_3$, que deveriam valer 0

λ_1 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **1**

λ_2 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **2**

λ_3 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **3**

x_i , y_i - Estimativas das coordenadas do robô no início da iteração i

θ_i - Estimativa da orientação do robô no início da iteração i

x_i^{novo} , y_i^{novo} - Estimativas das coordenadas do robô, obtidas no final da iteração i

θ_i^{novo} - Estimativa da orientação do robô, obtida no final da iteração i

Δx $\quad \Delta x = x_i^{novo} - x_i$

Δy $\quad \Delta y = y_i^{novo} - y_i$

$\Delta\theta$ $\quad \Delta\theta = \theta_i^{novo} - \theta_i$

Figura 4.10: Grandezas em jogo no Algoritmo de Triangulação Baseado no Método de Newton-Raphson.

A formulação geral (ampliada) do algoritmo admite a seguinte especificação (Anexo D):

1. Obter as estimativas iniciais de posição e de orientação;
2. Se a estimativa inicial conduz a situações de *overflow*, enviar erro e terminar.
3. Para o número desejado de iterações (até um máximo de **n**), fazer:

a) Obter **Δx**, **Δy** e **Δθ** resolvendo o sistema[8] (recorrendo à decomposição triangular LU):

$$\begin{bmatrix} \frac{y_1-y_i}{(x_1-x_i)^2} & \frac{1}{x_1-x_i} & -1-tg^2(\lambda_1+\theta_i) \\ \frac{y_2-y_i}{(x_2-x_i)^2} & \frac{1}{x_2-x_i} & -1-tg^2(\lambda_2+\theta_i) \\ \frac{y_3-y_i}{(x_3-x_i)^2} & \frac{1}{x_3-x_i} & -1-tg^2(\lambda_3+\theta_i) \end{bmatrix} \times \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} -\frac{y_1-y_i}{x_1-x_i}+tg(\lambda_1+\theta_i) \\ -\frac{y_2-y_i}{x_2-x_i}+tg(\lambda_2+\theta_i) \\ -\frac{y_3-y_i}{x_3-x_i}+tg(\lambda_3+\theta_i) \end{bmatrix}$$

b) Obter a nova estimativa da solução: $\begin{cases} x_i^{novo} \equiv x_i + \Delta x \\ y_i^{novo} \equiv y_i + \Delta y \\ \theta_i^{novo} \equiv \theta_i + \Delta\theta \end{cases}$

c) Terminar:

I. se a nova estimativa for suficientemente próxima da solução, ou seja, se:

$$\begin{cases} \left|\frac{\Delta x}{x_i^{novo}}\right| \le \varepsilon'_x \\ \left|\frac{\Delta y}{y_i^{novo}}\right| \le \varepsilon'_y \\ \left|\frac{\Delta\theta}{\theta_i^{novo}}\right| \le \varepsilon'_\theta \\ f_1(x,y,\theta) = \frac{y_1-(y_i+\Delta y)}{x_1-(x_i+\Delta x)} - tg\left[\lambda_1+(\theta_i+\Delta\theta)\right] \le \varepsilon \\ f_2(x,y,\theta) = \frac{y_2-(y_i+\Delta y)}{x_2-(x_i+\Delta x)} - tg\left[\lambda_2+(\theta_i+\Delta\theta)\right] \le \varepsilon \\ f_3(x,y,\theta) = \frac{y_3-(y_i+\Delta y)}{x_3-(x_i+\Delta x)} - tg\left[\lambda_3+(\theta_i+\Delta\theta)\right] \le \varepsilon \end{cases}$$

II. se houver indícios de divergência ou de condições conducentes a *overflow*.

III. se se atingir um número limite de iterações (**n**).

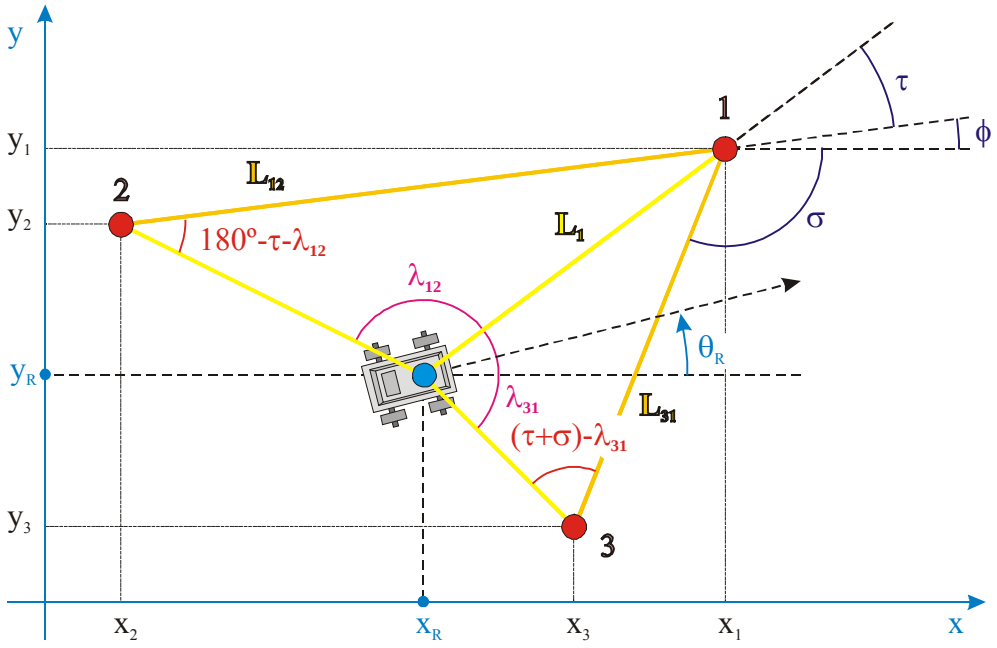
d) Fazer: $\begin{cases} x_i \equiv x_i^{novo} \\ y_i \equiv y_i^{novo} \\ \theta_i \equiv \theta_i^{novo} \end{cases}$

Para que o algoritmo se possa aplicar, é necessário que o robô disponha de estimativas iniciais da sua posição e da sua orientação. Estas podem obter-se por odometria, por exemplo. Se as estimativas não estiverem suficientemente próximas da solução, há divergência. Devido à natureza das equações não lineares utilizadas pelo algoritmo, não é fácil prever quando é que a divergência ocorre.

[8] O sistema não se pode resolver quando algum dos ângulos $\lambda_i+\theta$ (i=1, 2 ou 3) se torna igual a 90º ou 270º.

4.7 Algoritmo de Triangulação Geométrica

O Algoritmo de Triangulação Geométrica foi concebido por Yoram Koren (Cohen e Koss, 1992). É quase idêntico a um dos desenvolvidos por Mcgillem e Rappaport (1989) e muito semelhante ao proposto por Ji *et al.* (2003). Os três algoritmos baseiam-se nas propriedades de dois triângulos com um lado comum. Na Figura 4.11, o primeiro triângulo é formado pelo robô e pelas balizas 1 e 2. O segundo é formado pelo robô e as balizas 1 e 3. De cada triângulo conhece-se à partida apenas um lado (segmento que une as duas balizas) e mede-se, em cada instante, um dos ângulos (λ_{12} num dos triângulos e λ_{31} no outro). Para determinar o lado comum aos dois triângulos recorre-se à lei dos senos. O algoritmo de Ji *et al.* (2003) é facilmente adaptável à utilização com mais de três balizas. Mas possui a desvantagem de requerer a resolução de um sistema de equações correspondente à intersecção de duas das rectas que contém o robô e cada uma das balizas.



x_1 , y_1 - Coordenadas da baliza **1**
x_2 , y_2 - Coordenadas da baliza **2**
x_3 , y_3 - Coordenadas da baliza **3**
L_{12} - Distância entre as balizas **1** e **2**
L_{23} - Distância entre as balizas **2** e **3**
L_{31} - Distância entre as balizas **3** e **1**
L_1 - Distância entre o robô e a baliza **1**

λ_1 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **1**
λ_2 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **2**
λ_3 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **3**
λ_{12} - Ângulo formado pelos segmentos que unem o robô às balizas **1** e **2**
λ_{31} - Ângulo formado pelos segmentos que unem o robô às balizas **1** e **3**
x_R , y_R - Coordenadas do robô
θ_R - Orientação do robô

Figura 4.11: Grandezas em jogo no Algoritmo de Triangulação Geométrica.

O Algoritmo de Triangulação Geométrica é o seguinte:

1. Ordenar devidamente as balizas.
2. Seja $\lambda_{31} = 360^{o} + (\lambda_1 - \lambda_3)$
3. Seja $\lambda_{12} = \lambda_2 - \lambda_1$
4. Seja ϕ o ângulo entre o semieixo positivo dos xx e a linha formada pelos pontos das balizas 1 e 2.
5. Seja σ o ângulo entre o semieixo positivo dos xx e as balizas 1 e 3, mais o ângulo ϕ.
6. Seja $\gamma = \sigma - \lambda_{31}$
7. $p = \dfrac{L_{31} \cdot \text{sen}\lambda_{12}}{L_{12} \cdot \text{sen}\lambda_{31}}$
8. Seja $\tau = \text{arctg}\left[\dfrac{\text{sen}\lambda_{12} - p \cdot \text{sen}\gamma}{p \cdot \cos\gamma - \cos\lambda_{12}}\right]$
9. Seja $L_1 = \dfrac{L_{12} \cdot \text{sen}(\tau + \lambda_{12})}{\text{sen}\lambda_{12}}$
10. $x_R = x_1 - L_1 \cdot \cos(\phi + \tau)$
11. $y_R = y_1 - L_1 \cdot \sin(\phi + \tau)$
12. $\theta_R = \phi + \tau - \lambda_1$

Para além das restrições que são comuns a todos os algoritmos de autolocalização por triangulação, segundo Cohen e Koss (1992) o Algoritmo de Triangulação Geométrica possui também as seguintes restrições específicas:

a) As três balizas usadas pelo algoritmo têm de ser numeradas consecutivamente (1, 2 e 3) no sentido directo.

b) Tanto o ângulo entre as balizas 1 e 2 (λ_{12}) como o ângulo entre as balizas 1 e 3 (λ_{31}) têm de ser inferiores a 180º. Se isto não for verdade, as balizas têm de ser numeradas outra vez. A actual baliza 2 passa a ser a nova baliza 1 e as outras duas numeram-se no sentido directo. Esta operação é efectuada uma ou duas vezes, até que a condição enunciada esteja satisfeita.

Quando estas duas condições estão satisfeitas, as balizas dizem-se *devidamente ordenadas*.

A segunda condição implica que a localização não é possível quando o robô se encontra sobre o segmento de recta que une as balizas 1 e 2 ou sobre o segmento de recta que une as balizas 1 e 3. Mas, de facto, o algoritmo não funciona quando o robô se encontra sobre a recta definida pelas balizas 1 e 2 ou sobre a recta definida pelas balizas

1 e 3, pois nessas situações ocorrem divisões por zero (devido ao facto de algum dos ângulos λ_{12} ou λ_{31} ser igual a 0º ou 180º).

Como resultado de todas as restrições (tanto as gerais como as específicas), há zonas e percursos do plano de navegação nos quais o Algoritmo de Triangulação Geométrica não funciona.

Além disso, ainda de acordo com Cohen e Koss (1992), *"o algoritmo só funciona consistentemente quando o robô se encontra dentro do triângulo formado pelas três balizas. Há zonas fora do triângulo de balizas nas quais o algoritmo funciona, mas essas zonas são difíceis de determinar e são altamente dependentes do modo como se definem os ângulos"*[9]. Estes autores admitem que "*Não foi encontrada uma regra consistente para definir os ângulos por forma a garantir uma correcta execução deste método"*. De facto, no seu artigo não se especifica se os ângulos λ_1, λ_2 e λ_3 se medem no sentido directo ou no sentido inverso. Além disso, a definição dos ângulos ϕ e σ é ambígua:

- *"Seja ϕ* ***o ângulo*** *entre o semieixo positivo dos xx e a linha formada pelos pontos das balizas 1 e 2"*.

- *"Seja σ* ***o ângulo*** *formado pelo semieixo positivo dos xx com a recta que passa pelas balizas 1 e 3, mais o ângulo ϕ"*.

Acontece que duas rectas (ou segmentos de recta) concorrentes num ponto formam **dois ângulos** entre si. Cohen e Koss (1992) não esclarecem, para cada caso, qual dos ângulos se deve escolher para ϕ e para σ.

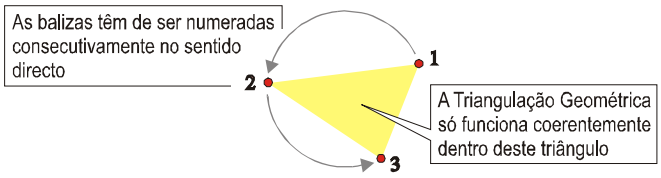


Figura 4.12: O algoritmo de Triangulação Geométrica só funciona consistentemente quando o robô se encontra dentro do triângulo formado por três balizas "devidamente ordenadas".

[9] Mcgillem e Rappaport (1989) também referem que, ao usar o seu algoritmo baseado na trigonometria, é necessário *"manter-se em contacto com os quadrantes dos ângulos para evitar importantes erros no cálculo da posição"*.

4.8 Algoritmo de Triangulação com Cálculo das Distâncias entre o Robô e as Balizas

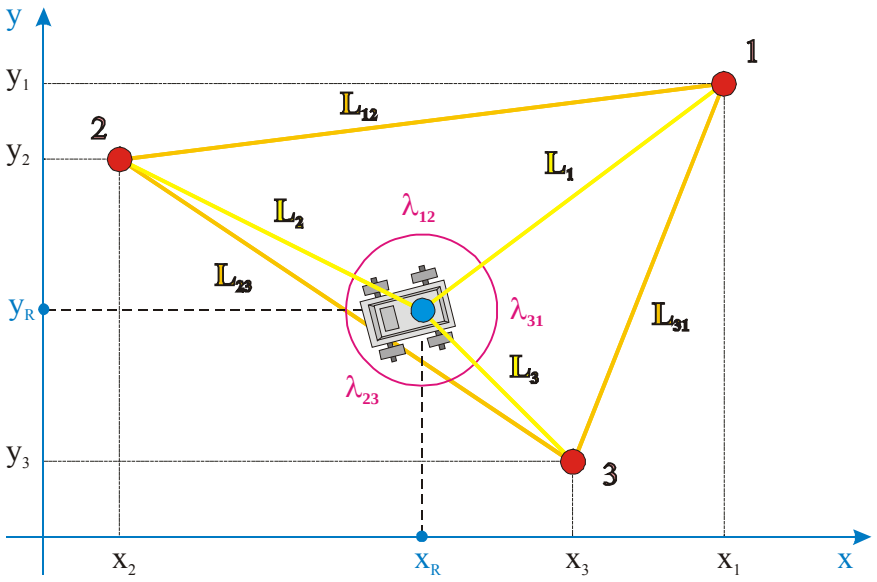
O algoritmo de Triangulação com Cálculo de Distâncias entre Robô e Balizas (Betke e Gurvits, 1997) consiste na resolução de dois sistemas de equações:

1. Para determinar $\mathbf{L_1}$, $\mathbf{L_2}$ e $\mathbf{L_3}$ (Figura 4.13), resolver (pelo método dos mínimos quadráticos, por exemplo) o seguinte sistema de equações não lineares (aplicação da lei dos cossenos):

$$\begin{cases} L_{12}^{\,2} = L_1^{\,2} + L_2^{\,2} - 2L_1L_2\cos\lambda_{12} \\ L_{23}^{\,2} = L_2^{\,2} + L_3^{\,2} - 2L_2L_3\cos\lambda_{23} \\ L_{31}^{\,2} = L_3^{\,2} + L_1^{\,2} - 2L_3L_1\cos\lambda_{31} \end{cases}$$

2. Para determinar $\mathbf{x_R}$ e $\mathbf{y_R}$ (Figura 4.13), resolver o seguinte sistema de equações lineares em **x** e **y** (Anexo E):

$$\begin{bmatrix} 2(x_2-x_1) & 2(y_2-x_1) \\ 2(x_3-x_1) & 2(y_3-x_1) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} L_1^{\,2} - L_2^{\,2} - x_1^{\,2} + x_2^{\,2} - y_1^{\,2} + y_2^{\,2} \\ L_1^{\,2} - L_3^{\,2} - x_1^{\,2} + x_3^{\,2} - y_1^{\,2} + y_3^{\,2} \end{bmatrix}$$



x_1 , y_1 - Coordenadas da baliza **1**
x_2 , y_2 - Coordenadas da baliza **2**
x_3 , y_3 - Coordenadas da baliza **3**
L_{12} - Distância entre as balizas **1** e **2**
L_{23} - Distância entre as balizas **2** e **3**
L_{31} - Distância entre as balizas **3** e **1**
L_1 - Distância entre o robô e a baliza **1**
L_2 - Distância entre o robô e a baliza **2**
L_3 - Distância entre o robô e a baliza **3**
λ_{12} - Ângulo formado pelos segmentos que unem o robô às balizas **1** e **2**
λ_{23} - Ângulo formado pelos segmentos que unem o robô às balizas **2** e **3**
λ_{31} - Ângulo formado pelos segmentos que unem o robô às balizas **1** e **3**
x_R , y_R - Coordenadas do robô

Figura 4.13: Grandezas em jogo no Algoritmo de Triangulação com Cálculo das Distâncias entre o Robô e as Balizas.

4.9 Algoritmo de Triangulação com Intersecção de Duas Circunferências

Duas balizas e o robô definem uma circunferência que passa pelos três pontos. Recorrendo a uma terceira baliza define-se outra circunferência que se intersecta com a primeira em dois pontos: num deles está a baliza comum às duas circunferências e no outro encontra-se o robô que se pretende localizar. Para determinar as coordenadas desses pontos no referencial x0y (Figura 4.14) é possível recorrer ao seguinte algoritmo, parcialmente usado por Stella *et al.* (1995) e também por Mcgillem e Rappaport (1989)[10]:

1. $R_A = \dfrac{L_{12}}{2 \cdot \operatorname{sen} \lambda_{12}} \qquad (\lambda_{12} \neq 0^o \ \wedge \ \lambda_{12} \neq 180^o)$

2. $R_B = \dfrac{L_{31}}{2 \cdot \operatorname{sen} \lambda_{31}} \qquad (\lambda_{31} \neq 0^o \ \wedge \ \lambda_{31} \neq 180^o)$

3. Para determinar $\mathbf{x_{CA}}$ e $\mathbf{y_{CA}}$, resolver o seguinte sistema de equações não lineares em **x** e **y**:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 = {R_A}^2 \\ (x - x_2)^2 + (y - y_2)^2 = {R_A}^2 \end{cases}$$

4. Para determinar $\mathbf{x_{CB}}$ e $\mathbf{y_{CB}}$, resolver o seguinte sistema de equações não lineares em **x** e **y**:

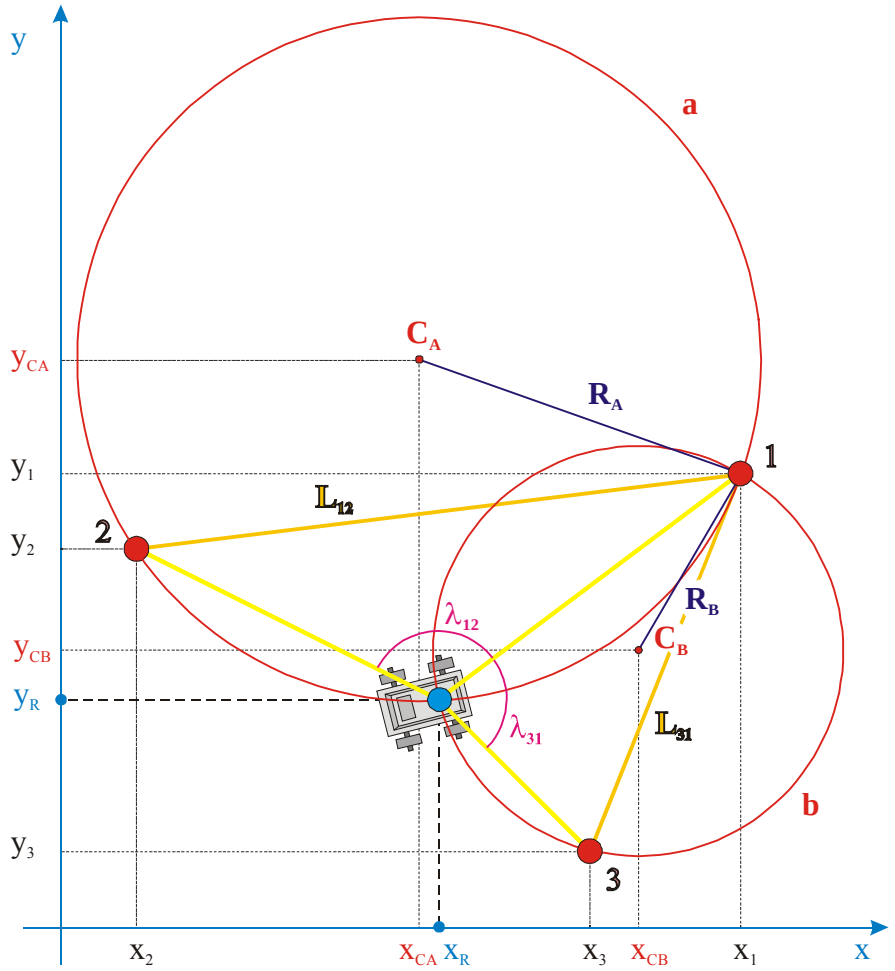
$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 = {R_B}^2 \\ (x - x_3)^2 + (y - y_3)^2 = {R_B}^2 \end{cases}$$

5. Para determinar $\mathbf{x_R}$ e $\mathbf{y_R}$, resolver o seguinte sistema de equações não lineares em **x** e **y**:

$$\begin{cases} (x - x_{CA})^2 + (y - y_{CA})^2 = {R_A}^2 \\ (x - x_{CB})^2 + (y - y_{CB})^2 = {R_B}^2 \end{cases}$$

Este algoritmo requer a resolução de três sistemas de equações não lineares.

[10] Stella *et al.* (1995) não especificam o modo de calcular x_R e y_R, uma vez determinadas as coordenadas dos centros das duas circunferências. Mcgillem e Rappaport (1989) calculam as coordenadas dos centros das duas circunferências sem recorrer à resolução de sistemas de equações.



x_1 , y_1	- Coordenadas da baliza **1**	R_A	- Raio da circunferência **a**
x_2 , y_2	- Coordenadas da baliza **2**	R_B	- Raio da circunferência **b**
x_3 , y_3	- Coordenadas da baliza **3**	C_A	- Centro da circunferência **a**
L_{12}	- Distância entre as balizas **1** e **2**		Coordenadas: x_{CA} , y_{CA}
L_{31}	- Distância entre as balizas **3** e **1**	C_B	- Centro da circunferência **b**
λ_{12}	- Ângulo formado pelos segmentos que unem o robô às balizas **1** e **2**		Coordenadas: x_{CB} , y_{CB}
λ_{31}	- Ângulo formado pelos segmentos que unem o robô às balizas **1** e **3**	x_R , y_R	- Coordenadas do robô

Figura 4.14: Grandezas em jogo no Algoritmo de Triangulação com Intersecção de Duas Circunferências.

Há duas soluções matematicamente possíveis para cada um dos sistemas das linhas 3 e 4 do algoritmo. De facto, existem duas circunferências de raio R_A que passam pelas balizas 1 e 2, e cada uma dessas circunferências possui o seu centro. E existem duas circunferências de raio R_B, cada uma com o seu centro, que passam pelas balizas 1 e 3. Este facto dá origem a ambiguidade na determinação da posição do robô (Figura 4.1 e Figura 4.2).

Há também duas soluções matematicamente possíveis para o sistema da linha 5 do algoritmo, que correspondem às coordenadas da baliza 1 e às coordenadas do robô.

As duas circunferências ficam coincidentes quando o robô se encontra sobre a circunferência definida pelas três balizas. Nessa situação, as duas equações do sistema da linha 5 do algoritmo tornam-se idênticas e não é possível obter uma solução.

A posição do robô também não pode ser calculada quando este se encontra sobre a recta que passa pelas balizas 1 e 2 ou sobre a recta que passa pelas balizas 1 e 3. R_A torna-se infinito no primeiro caso e R_B torna-se infinito no segundo.

Mcgillem e Rappaport (1989) calculam as coordenadas dos centros das duas circunferências sem recorrer à resolução de sistemas de equações. No entanto, os resultados obtidos com as expressões propostas por estes autores dependem da forma como se definem os vários ângulos utilizados pelo algoritmo. Sem uma cuidadosa definição de ângulos – que não é apresentada pelos autores – as expressões não produzem resultados correctos em todas as regiões do plano de navegação.

Após alguma manipulação de expressões e uma substituição de variáveis, a resolução do sistema de equações da linha 5 pode conduzir à resolução de uma equação do segundo grau numa das variáveis, x ou y. Como se conhece uma das soluções da equação (uma das coordenadas da baliza 1), é possível decompor essa equação em dois factores e, finalmente, obter a solução desconhecida resolvendo uma simples equação do primeiro grau. Mcgillem e Rappaport (1989) recorrem a este processo e apresentam um conjunto de equações lineares que, usadas em substituição do sistema da linha 5 do algoritmo, permitem calcular a posição do robô sem recorrer à resolução de sistemas de equações.

O algoritmo que se obtém depois de feitas as duas modificações referidas torna-se muito semelhante ao Algoritmo de Triangulação com Intersecção Geométrica de Circunferências – apresentado de seguida – que também não requer a resolução de sistemas de equações.

4.10 Algoritmo de Triangulação com Intersecção Geométrica de Circunferências

Cohen e Koss (1992) apresentam o seguinte algoritmo (Figura 4.15):

1. Ordenar adequadamente as balizas. [11]
2. $\lambda_{12} = \lambda_2 - \lambda_1$
   Se λ_{12} for demasiado pequeno ou igual a 90º ou 270º, enviar mensagem de erro pois ocorrerá uma divisão por 0.
3. $\lambda_{23} = \lambda_3 - \lambda_2$
   Se λ_{23} for demasiado pequeno ou igual a 90º ou 270º, enviar mensagem de erro pois ocorrerá uma divisão por 0.
4. $R_A = \dfrac{L_{12}}{2 \cdot \operatorname{sen} \lambda_{12}} \qquad (\lambda_{12} \neq 0^o \ \wedge \ \lambda_{12} \neq 180^o)$
5. $R_B = \dfrac{L_{23}}{2 \cdot \operatorname{sen} \lambda_{23}} \qquad (\lambda_{23} \neq 0^o \ \wedge \ \lambda_{23} \neq 180^o)$
6. $\overline{C_A M_{12}} = \dfrac{L_{12}}{2 \cdot \operatorname{tg} \lambda_{12}}$
7. $\overline{C_B M_{23}} = \dfrac{L_{23}}{2 \cdot \operatorname{tg} \lambda_{23}}$
8. Sejam v_{12x} e v_{12y} as componentes de um vector unitário orientado da baliza **1** para a baliza **2**.
9. Sejam v_{23x} e v_{23y} as componentes de um vector unitário orientado da baliza **2** para a baliza **3**.
10. $x_{CA} = x_{M12} - \left(\overline{C_A M_{12}}\right) \cdot v_{12y}$
11. $y_{CA} = y_{M12} + \left(\overline{C_A M_{12}}\right) \cdot v_{12x}$
12. $x_{CB} = x_{M23} - \left(\overline{C_B M_{23}}\right) \cdot v_{23y}$
13. $y_{CB} = y_{M23} + \left(\overline{C_B M_{23}}\right) \cdot v_{23x}$
14. Enviar mensagem de erro se os centros das duas circunferências estiverem muito próximos.
15. Enviar mensagem de erro se γ for muito grande. [12]
16. Sejam v_{BAx} e v_{BAy} as componentes de um vector unitário orientado de $\mathbf{C_B}$ para $\mathbf{C_A}$.
17. $L_2 = 2 \cdot R_B \cdot \operatorname{sen} \gamma$ [12, 13]
18. $\overline{C_B M_2} = R_B \cdot \cos \gamma$ [12]
19. $x_R = 2 \cdot x_{M2} - x_2 + 0{,}5$ [14]
20. $y_R = 2 \cdot y_{M2} - y_2 + 0{,}5$ [15]
21. $\phi = \operatorname{arctg}\left(\dfrac{y_1 - y_R}{x_1 - x_R}\right)$; ϕ é a orientação da baliza **1** a partir da posição verdadeira do robô.
22. Se $\phi > 0^o$ então: $\theta_R = \phi - \lambda_1$ [16]
    senão: $\theta_R = \phi - \lambda_1 + 360^o$ [16]
23. Enviar resultado.

---

[11] Os autores referem que esta ordenação é idêntica à requerida na Triangulação Geométrica.

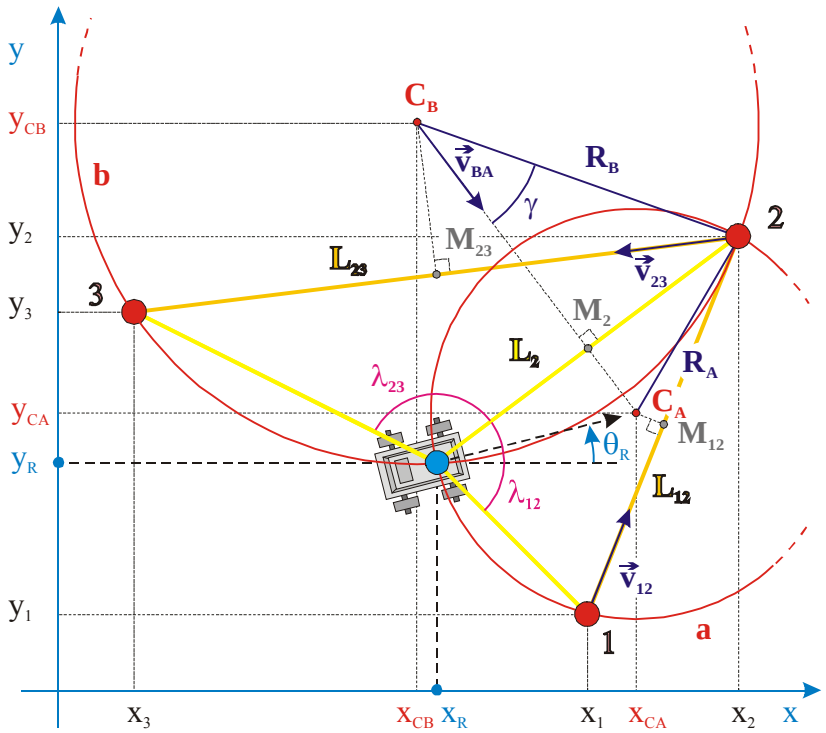
[12] Não se apresenta o cálculo de γ. E este passo é equivalente à mensagem de erro do passo anterior se *"γ muito grande"* significa *"γ próximo de 90º"* $\left(\overline{C_A C_B} \to 0 \ \Rightarrow \ \gamma \to 90^o\right)$.

[13] Os autores não dizem para que serve este passo.

[14] Não se apresenta o cálculo de $\mathbf{x_{M2}}$ nem a razão pela qual se soma 0,5 ao resultado obtido.

[15] Não se apresenta o cálculo de $\mathbf{y_{M2}}$ nem a razão pela qual se soma 0,5 ao resultado obtido.

[16] Os autores chamam *"erro de orientação"* e não $\boldsymbol{\theta_R}$ a esta quantidade.



x_1 , y_1 - Coordenadas da baliza **1**

x_2 , y_2 - Coordenadas da baliza **2**

x_3 , y_3 - Coordenadas da baliza **3**

L_{12} - Distância entre as balizas **1** e **2**

L_{23} - Distância entre as balizas **2** e **3**

$\vec{v}_{12}$ - Vector unitário orientado da baliza **1** para a baliza **2**
Componentes: v_{12x} , v_{12y}

$\vec{v}_{23}$ - Vector unitário orientado da baliza **2** para a baliza **3**
Componentes: v_{23x} , v_{23y}

M_{12} - Ponto médio da corda situada entre as balizas **1** e **2**
Coordenadas: x_{M12} , y_{M12}

M_{23} - Ponto médio da corda situada entre as balizas **2** e **3**
Coordenadas: x_{M23} , y_{M23}

λ_1 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **1**

λ_2 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **2**

λ_3 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **3**

λ_{12} - Ângulo formado pelos segmentos que unem o robô às balizas **1** e **2**

λ_{23} - Ângulo formado pelos segmentos que unem o robô às balizas **2** e **3**

L_2 - Distância entre o robô e a baliza **2**

M_2 - Ponto médio da corda situada entre o robô e a baliza **2**
Coordenadas: x_{M2} , y_{M2}

R_A - Raio da circunferência **a**

R_B - Raio da circunferência **b**

C_A - Centro da circunferência **a**
Coordenadas: x_{CA} , y_{CA}

C_B - Centro da circunferência **b**
Coordenadas: x_{CB} , y_{CB}

$\vec{v}_{BA}$ - Vector unitário orientado do ponto $\mathbf{C_B}$ para o ponto $\mathbf{C_A}$
Componentes: v_{BAx} , v_{BAy}

$\overline{C_AM_{12}}$ - Distância entre os pontos $\mathbf{C_A}$ e $\mathbf{M_{12}}$
(Em rigor, não é adequado chamar "distância" a esta quantidade, que pode ser negativa.)

$\overline{C_BM_{23}}$ - Distância entre os pontos $\mathbf{C_B}$ e $\mathbf{M_{23}}$
(Em rigor, não é adequado chamar "distância" a esta quantidade, que pode ser negativa.)

$\overline{C_BM_2}$ - Distância entre os pontos $\mathbf{C_B}$ e $\mathbf{M_2}$

ϕ - Orientação da baliza **1** a partir da posição verdadeira do robô.

x_R , y_R - Coordenadas do robô

θ_R - Orientação do robô

Figura 4.15: Grandezas em jogo no Algoritmo de Triangulação com Intersecção Geométrica de Circunferências.

À semelhança do algoritmo anteriormente descrito, este também considera duas circunferências que se intersectam na baliza 1 e no ponto em que se encontra o robô. A posição e a orientação do robô são determinadas sem a resolução de sistemas de equações.

As duas circunferências ficam coincidentes quando o robô se encontra sobre a circunferência definida pelas três balizas e nessa situação não é possível obter uma solução (esta situação está prevista na linha 14 do algoritmo).

O robô também não se pode localizar quando se encontra sobre a recta que passa pelas balizas 1 e 2 ou sobre a recta que passa pelas balizas 2 e 3. R_A torna-se infinito no primeiro caso e R_B torna-se infinito no segundo.

Cohen e Koss (1992) referem ainda a impossibilidade de o robô se localizar se algum dos ângulos λ_{12} ou λ_{31} for igual a 90º ou 270º. No entanto, nessas circunstâncias ocorrem apenas divisões por infinito (e não divisões por zero, como se indica no algoritmo) que geralmente não impedem a correcta execução do programa que implementa o algoritmo[17].

Segundo Cohen e Koss (1992) *"ϕ é a orientação da baliza **1** a partir da posição verdadeira do robô"*. No entanto, o valor de ϕ calculado na linha 21 está sempre compreendido entre –90º e 90º, mesmo quando $90º < \phi < 270º$. Há por isso uma ambiguidade de 180º na orientação calculada do robô.

[17] Isto é verdade, pelo menos, quando se recorre à linguagem de programação Java 2.

4.11 Algoritmo de Triangulação com Intersecção de Três Circunferências

Se houver três balizas disponíveis é possível considerar três circunferências, cada uma das quais definida pelos pontos nos quais se encontram o robô e duas balizas. As três circunferências intersectam-se apenas no ponto em que o robô se encontra. As coordenadas desse ponto no referencial x0y (Figura 4.16) podem determinar-se recorrendo ao seguinte algoritmo, utilizado por Fuentes *et al.* (1995):

3. $R_A = \dfrac{L_{12}}{2 \cdot \operatorname{sen} \lambda_{12}} \qquad (\lambda_{12} \neq 0^o \ \wedge \ \lambda_{12} \neq 180^o)$

4. $R_B = \dfrac{L_{23}}{2 \cdot \operatorname{sen} \lambda_{23}} \qquad (\lambda_{23} \neq 0^o \ \wedge \ \lambda_{23} \neq 180^o)$

5. $R_C = \dfrac{L_{31}}{2 \cdot \operatorname{sen} \lambda_{31}} \qquad (\lambda_{31} \neq 0^o \ \wedge \ \lambda_{31} \neq 180^o)$

6. Determinar $\mathbf{x_{CA}}$ e $\mathbf{y_{CA}}$, tendo em conta que:

   a. se $\lambda_{12}<90^o$ então

   $$\begin{cases} A_X = R_A \cdot \operatorname{sen}(\lambda_{12} + 90^o - \eta) \\ A_Y = \sqrt{R_A^2 - A_X^2} \end{cases}$$

   b. se $\lambda_{12}>90^o$ então

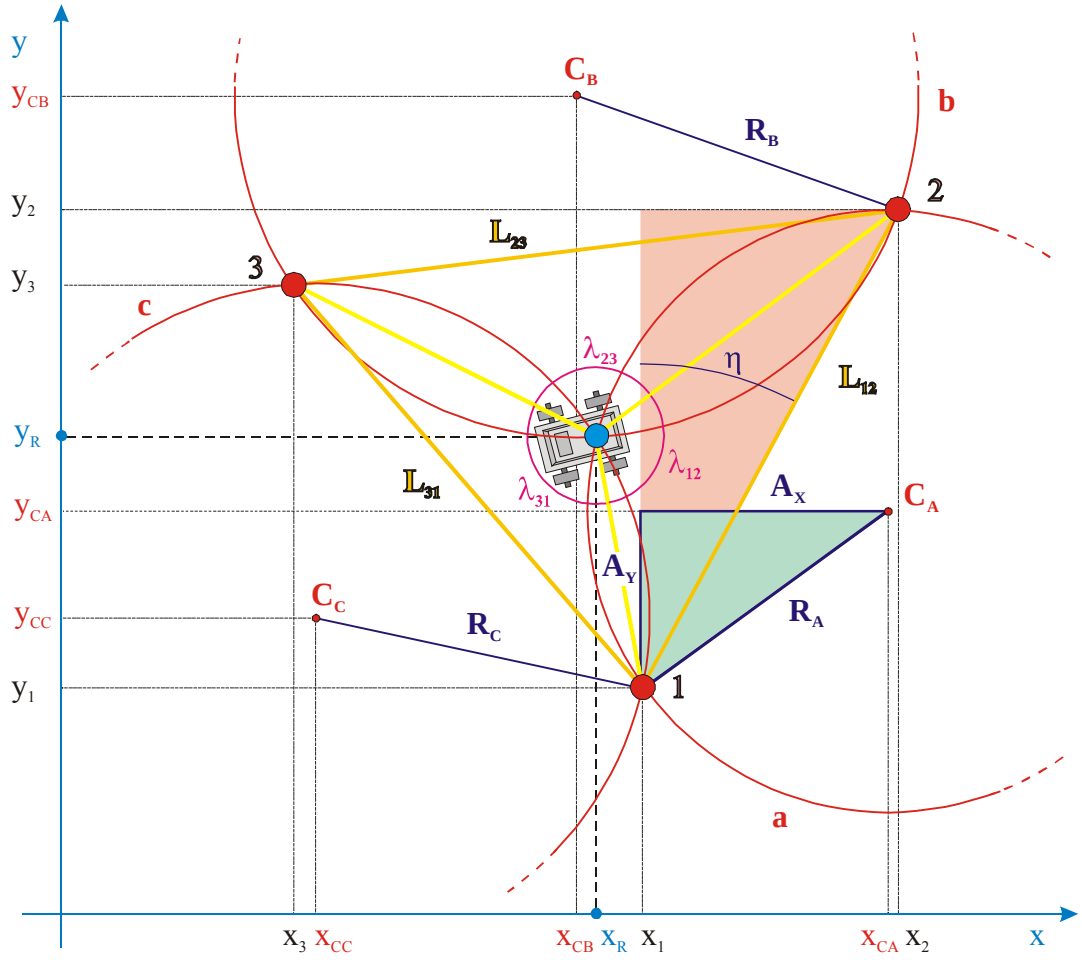
   $$\begin{cases} A_X = R_A \cdot \operatorname{sen}(\lambda_{12} - 90^o - \eta) \\ A_Y = \sqrt{R_A^2 - A_X^2} \end{cases}$$

7. Determinar $\mathbf{x_{CB}}$, $\mathbf{y_{CB}}$, $\mathbf{x_{CC}}$ e $\mathbf{y_{CC}}$ recorrendo ao processo utilizado para determinar $\mathbf{x_{CA}}$ e $\mathbf{y_{CA}}$.

8. Para determinar $\mathbf{x_R}$ e $\mathbf{y_R}$ resolver o seguinte sistema de equações lineares em **x** e **y** (Anexo F):

$$\begin{bmatrix} 2 \cdot (x_{CA} - x_{CB}) & 2 \cdot (y_{CA} - y_{CB}) \\ 2 \cdot (x_{CB} - x_{CC}) & 2 \cdot (y_{CB} - y_{CC}) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \left(x_{CA}^2 - x_{CB}^2\right) + \left(y_{CA}^2 - y_{CB}^2\right) - \left(R_A^2 - R_B^2\right) \\ \left(x_{CB}^2 - x_{CC}^2\right) + \left(y_{CB}^2 - y_{CC}^2\right) - \left(R_B^2 - R_C^2\right) \end{bmatrix}$$

As três circunferências ficam coincidentes quando o robô se encontra sobre a circunferência definida pelas três balizas. Como os seus três raios são iguais e os seus três centros possuem as mesmas coordenadas, não é possível obter uma solução com o sistema de equações indicado na linha 8 do algoritmo.



x_1 , y_1 - Coordenadas da baliza **1**
x_2 , y_2 - Coordenadas da baliza **2**
x_3 , y_3 - Coordenadas da baliza **3**
L_{12} - Distância entre as balizas **1** e **2**
L_{23} - Distância entre as balizas **2** e **3**
L_{31} - Distância entre as balizas **3** e **1**
λ_{12} - Ângulo formado pelos segmentos que unem o robô às balizas **1** e **2**
λ_{23} - Ângulo formado pelos segmentos que unem o robô às balizas **2** e **3**
λ_{31} - Ângulo formado pelos segmentos que unem o robô às balizas **1** e **3**

R_A - Raio da circunferência **a**
R_B - Raio da circunferência **b**
R_C - Raio da circunferência **c**
C_A - Centro da circunferência **a**
Coordenadas: x_{CA} , y_{CA}
C_B - Centro da circunferência **b**
Coordenadas: x_{CB} , y_{CB}
C_C - Centro da circunferência **c**
Coordenadas: x_{CC} , y_{CC}
x_R , y_R - Coordenadas do robô

Figura 4.16: Grandezas em jogo no Algoritmo de Triangulação com Intersecção de Três Circunferências.

A posição do robô também não pode ser calculada quando este se encontra sobre alguma das três rectas que passam simultaneamente por duas balizas porque, nessas circunstâncias, um dos raios R_A, R_B ou R_C torna-se infinito[18].

[18] Nos algoritmos que recorrem à intersecção de apenas duas circunferências, o robô geralmente só está impossibilitado de se localizar sobre duas destas rectas.

4.12 Algoritmo de Triangulação do *Imperial College Beacon Navigation System*

Este algoritmo é referido por Everett (1995) e Borenstein *et al.* (1996):

1. $\beta = \text{arctg}\left[\dfrac{2\cdot \text{tg}\lambda_{12}\cdot \text{tg}\lambda_{23}}{\text{tg}\lambda_{12} - \text{tg}\lambda_{23}}\right] - \lambda_{12}$ [19]

2. $L_3 = \dfrac{L\cdot \text{sen}(\beta + \lambda_{23})}{\text{sen}\lambda_{23}}$

3. $\tau = \phi - \beta$

4. $x_R = x_3 + L_3 \cdot \cos\tau$

5. $y_R = y_3 + L_3 \cdot \text{sen}\tau$

A principal limitação do algoritmo reside no facto de só funcionar se as três balizas forem colineares e estiverem igualmente espaçadas, como se observa na Figura 4.17.

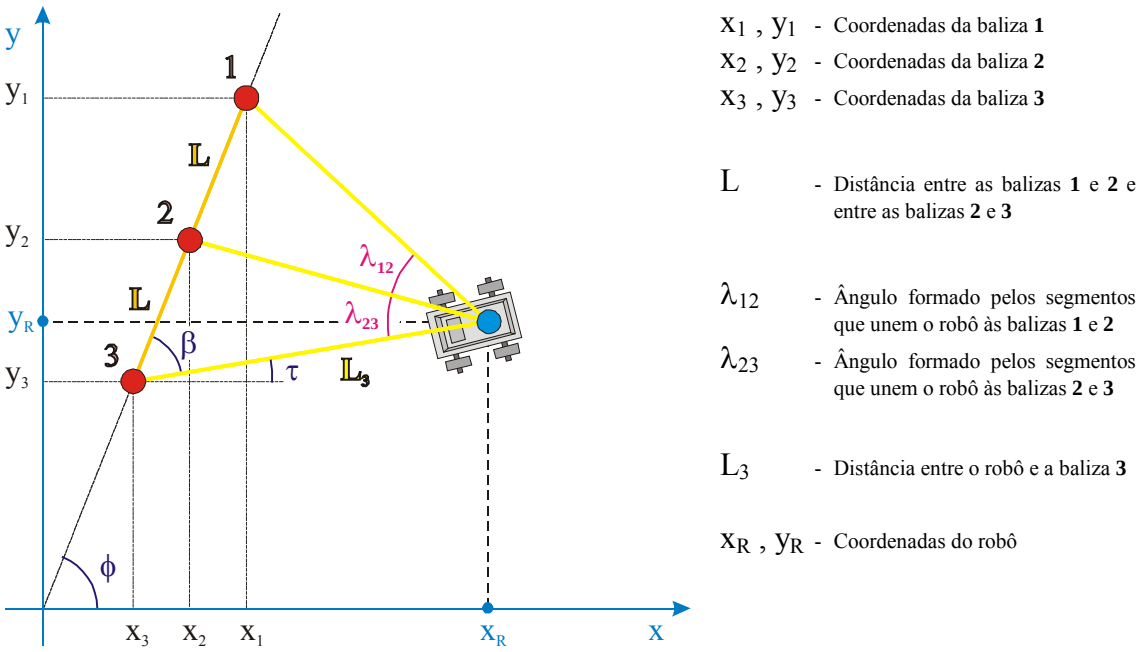


Figura 4.17: Grandezas em jogo no Algoritmo de Triangulação do *Imperial College Beacon Navigation System.*

[19] Borenstein *et al.* (1996) referem, para o cálculo de β, a seguinte expressão (que não funciona):

$$\beta = \text{arctg}\left[\frac{2\cdot \text{tg}\lambda_{12}\cdot \text{tg}\lambda_{23}}{\text{tg}\lambda_{12} - \text{tg}\lambda_{23}} - 1\right]$$

4.13 Conclusões

Foram analisados vários algoritmos de triangulação que se podem utilizar na autolocalização absoluta a duas dimensões de robôs móveis, com sistemas passivos de localização.

No ponto 4.1 apresentou-se uma definição do problema da autolocalização absoluta por triangulação.

No ponto 4.2 abordou-se a ambiguidade de posição que consiste no facto de, a um dado conjunto de medidas de ângulos, corresponder mais do que uma posição possível no plano de navegação. Apresentaram-se algumas das soluções habitualmente utilizadas para resolver este problema, que envolvem o conhecimento prévio de uma região do plano de navegação na qual um robô está constrangido a navegar ou o conhecimento da sua posição aproximada, obtida por um método de localização diferente da triangulação.

No ponto 4.3 analisaram-se as duas restrições que são comuns a todos os algoritmos de autolocalização baseados exclusivamente na triangulação:

1. Um robô tem de "ver", pelo menos, três balizas para se poder localizar;

2. Um robô não se consegue localizar quando está sobre a circunferência definida por três balizas não colineares ou a recta definida por três balizas colineares.

Resultando destas duas restrições, há algumas linhas do plano de navegação nas quais a autolocalização por triangulação não é possível.

Além das duas restrições apontadas, cada um dos algoritmos estudados nos pontos 4.4 a 4.12, utilizados ou referidos no âmbito da robótica móvel, possui limitações específicas. As principais encontram-se resumidas na (Tabela 4.1). Alguns destes algoritmos não incluem o cálculo da orientação do robô. Não se trata de uma limitação importante, uma vez que a orientação se pode calcular facilmente uma vez determinada a posição, a partir da medida do ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô a uma das balizas.

Tabela 4.1: Principais limitações específicas dos algoritmos de triangulação com três balizas estudados.

	Algoritmo Simples de Triangulação	Algoritmo de Triangulação Baseado na Pesquisa Iterativa	Algoritmo de Triangulação Baseado no Método de Newton-Raphson	Algoritmo de Triangulação Geométrica	Algoritmo de Triangulação com Cálculo das Distâncias Entre o Robô e as Balizas	Algoritmo de Triangulação com Intersecção de Duas Circunferências	Algoritmo de Triangulação com Intersecção Geométrica de Circunferências	Algoritmo de Triangulação com Intersecção de Três Circunferências	Algoritmo de Triangulação do *Imperial College B.N.S.*
Requer ordenação de balizas				●			●		
Requer uma particular configuração de balizas									●
Requer estimativas iniciais de posição e de orientação			●						
É muito mais lento que outros algoritmos		●							
Só funciona coerentemente dentro do triângulo formado por três balizas				●					
Não funciona se o robô estiver sobre parte ou a totalidade das rectas definidas por cada par de balizas						●	●	●	
Requer a resolução de sistemas de equações	●		●		●	● (só na versão original)		●	
Produz soluções múltiplas	●	●				● (só na versão original)	●		
Nem sempre produz uma solução, mesmo para entradas válidas			●						

O Algoritmo de Triangulação Geométrica (ponto 4.7) tem sido considerado de menor interesse. Mcgillem e Rappaport (1989) referem que, ao usar o seu algoritmo baseado na trigonometria (muito semelhante ao Algoritmo de Triangulação Geométrica), é necessário *"manter-se em contacto com os quadrantes dos ângulos para evitar importantes erros no cálculo da posição"*. Segundo Cohen e Koss (1992), o Algoritmo de Triangulação Geométrica é rápido mas *"só funciona consistentemente quando o robô se encontra dentro do triângulo formado pelas três balizas"*. Estes autores afirmam que *"Não foi encontrada uma regra consistente para definir os ângulos por forma a garantir uma correcta execução deste método"*.

Tanto Mcgillem e Rappaport (1989) como Cohen e Koss (1992) preferem algoritmos baseados na intersecção de circunferências. No entanto, estes algoritmos possuem uma limitação que lhes é inerente e não pode ser eliminada: o raio da circunferência definida pelo robô e por duas balizas torna-se infinito quando as balizas ficam colineares com o robô (a circunferência degenera numa recta) e, nessas circunstâncias, a posição do robô não pode ser calculada. Assim, nos algoritmos que recorrem à intersecção de duas circunferências há duas rectas do plano de navegação, cada uma delas definida por um par de balizas, ao longo das quais a localização não é possível. No Algoritmo de Triangulação com Intersecção de Três Circunferências (ponto 4.11) a localização não é possível ao longo das três rectas definidas por cada par de balizas.

No Capítulo 6 mostrar-se-á que é possível generalizar o Algoritmo de Triangulação Geométrica de forma a que este funcione, sem precisar de ordenação de balizas, fora do triângulo formado por três balizas e, em particular, ao longo das rectas mencionadas no parágrafo anterior. À partida, este algoritmo é rápido, simples e versátil, uma vez que não recorre à resolução de sistemas de equações e não requer nenhuma particular configuração de balizas. Os testes realizados por Cohen e Koss (1992) revelam que o Algoritmo de Triangulação Geométrica é o mais rápido dos quatro algoritmos analisados nos pontos 4.5, 4.6, 4.7 e 4.10. De acordo com esses testes, verifica-se que:

- o Algoritmo de Triangulação com Intersecção Geométrica de Circunferências (ponto 4.10) é cerca de 1,3 vezes mais lento;

- o Algoritmo de Triangulação Baseado no Método de Newton-Raphson (ponto 4.6) é cerca de 2,5 vezes mais lento;

- o Algoritmo de Triangulação Baseado na Pesquisa Iterativa (ponto 4.5) é cerca de 873,2 vezes mais lento.

O Algoritmo de Triangulação Geométrica também não requer estimativas iniciais de posição ou de orientação, não produz soluções múltiplas e o problema da solução divergir não se coloca (não é um método iterativo). As características mencionadas justificam o trabalho realizado para eliminar as suas limitações específicas.

Neste capítulo deram-se ainda os seguintes contributos:

- No ponto 4.4 sugeriu-se um algoritmo de triangulação simples, mas que consiste na resolução de um sistema de três equações não lineares (os algoritmos de outros autores, apresentados nos pontos 4.5 a 4.12, são mais complexos mas envolvem cálculos mais fáceis de executar).

- No ponto 4.5 sugeriu-se uma especificação do Algoritmo de Triangulação Baseado na Pesquisa Iterativa.

- No ponto 4.6 sugeriu-se uma especificação do Algoritmo de Triangulação Baseado no Método de Newton-Raphson.

5. Quadro de Análise da Autolocalização Absoluta por Triangulação com Três Balizas

Neste capítulo propõe-se um quadro de análise da autolocalização absoluta por triangulação com três balizas[1]. Constitui a base da generalização do Algoritmo de Triangulação Geométrica que será apresentada no Capítulo 6, mas é aplicável a outros algoritmos. Inclui um método capaz de caracterizar em tempo real as incertezas associadas à posição e à orientação calculadas e de detectar situações nas quais a localização não é possível. Tais capacidades são necessárias à aplicabilidade de qualquer método de localização e à *integridade*[2] do sistema que o implemente.

O primeiro passo consiste numa cuidadosa definição de ângulos a utilizar em algoritmos de triangulação. Em concreto, nos pontos 5.1 a 5.3 definem-se os ângulos necessários para

- caracterizar a configuração de balizas;
- determinar sem ambiguidade a posição e a orientação do robô.

De acordo com estas definições de ângulos, no ponto 5.4 sugere-se uma nova especificação do problema da autolocalização absoluta a duas dimensões por triangulação.

No ponto 5.5 aborda-se a relação entre a posição do robô e os ângulos λ_{12} e λ_{31}, usados como variáveis de entrada em algoritmos de cálculo de posição por triangulação. A compreensão desta relação é fundamental para a concepção e correcta implementação do método de caracterização de incertezas apresentado no ponto 5.6.

Em geral, a posição calculada mediante um determinado algoritmo não coincide com a posição verdadeira do robô e a orientação calculada também é diferente da sua

[1] Parte-se do princípio que todas as balizas utilizadas neste capítulo são emissoras, omnidireccionais, pontuais, distinguíveis, com padrão de emissão isotrópico e alcance infinito.

[2] *"Integridade: capacidade de um sistema de fornecer atempadamente avisos aos utilizadores quando o sistema não deve ser usado para a navegação"* (DoD, 2001).

orientação verdadeira. Os erros de medição, entre outros, dão origem a um *erro de posição* (uma distância) e também a um *erro de orientação* (um ângulo). Na autolocalização por triangulação com balizas, os erros de medição de ângulos constituem, geralmente, a principal fonte dos erros de posição e de orientação, cujos valores também dependem da posição do robô relativamente às balizas (esta afecta a sensibilidade dos algoritmos de localização aos erros de medição). O valor exacto de um erro de medição é sempre desconhecido, mas geralmente é possível associar a cada medida uma *incerteza de medição* que dá origem a incertezas de posição e de orientação. Na prática, a posição e a orientação calculadas são inúteis para efeitos de navegação quando não se fazem acompanhar das respectivas incertezas.

Diversos autores classificam os métodos de localização de robôs móveis como sendo *probabilísticos* ou *de erros limitados,* de acordo com o modo como abordam a análise de incertezas (Hager *et al.,* 1993; Di Marco *et al.,* 2000; Garulli e Vicino, 2001; Gutmann, 2002; Ashokaraj *et al.,* 2003; Gning e Bonnifait, 2004):

- Nos *métodos probabilísticos* (Betke e Gurvits, 1997; Gutmann, 2002; Shimshoni, 2002; Kleeman, 2003; Lee *et al.*, 2003; Prasser e Wyeth, 2003; Costa *et al.*, 2004) parte-se do princípio que os erros possuem determinadas distribuições de probabilidade – geralmente, não limitadas – e utilizam-se representações estatísticas explícitas dos erros.

  A localização a duas dimensões é encarada como o processo de determinar a probabilidade de o robô se encontrar num ponto ou numa região do plano de navegação e de a sua orientação ter um certo valor ou então um valor contido num determinado intervalo

  Estes métodos utilizam quase sempre técnicas de fusão sensorial que costumam requerer a aproximação de funções não lineares por funções lineares. Para garantir a validade dessa aproximação, é necessário que as incertezas associadas aos erros se mantenham abaixo de um determinado limiar.

- Nos *métodos de erros limitados* parte-se do princípio que todos os erros que dão origem aos erros e posição e de orientação são limitados em grandeza, com limites conhecidos.

  Sem fazer suposições sobre as distribuições de probabilidade desses erros, muitos autores recorrem à Análise de Intervalos – baseada na Teoria de Conjuntos – para determinar um conjunto de regiões que, garantidamente, contêm a posição verdadeira e a orientação verdadeira do robô, de acordo com a informação disponível (Di Marco *et al.,* 2000; Garulli e Vicino, 2001; Meizel *et al.,* 2002; Jaulin *et al.,* 2002; Ashokaraj *et al.,* 2003; Briechle e Hanebeck, 2004; Gning e Bonnifait, 2004). Sutherland (1994) e Sutherland e Thompson (1994) não recorrem à Teoria de Conjuntos[3] e também definem regiões com estas características. Mas consideram subdivisões dessas regiões e, partindo do princípio que os erros de medição possuem uma distribuição de probabilidade rectangular[4], verificam qual é a subdivisão com maior probabilidade de conter a posição verdadeira.

  Os métodos de erros limitados costumam ser mais fáceis de implementar que os métodos probabilísticos quando as equações usadas são não lineares – como é o caso da autolocalização por triangulação. Alguns não requerem a aproximação dessas funções por funções lineares.

  Em sistemas com muitas dimensões, os métodos probabilísticos aplicam-se com maior facilidade que os métodos de erros limitados.

No ponto 5.6 propõe-se um novo método que permite, em tempo real, caracterizar as incertezas associadas à posição e à orientação calculadas e também detectar situações nas quais a localização não é possível. Pode ser usado para

[3] Sutherland (1994) e Sutherland e Thompson (1994) apresentam um método de autolocalização absoluta, a duas dimensões, por triangulação com três balizas, no qual se supõe que os erros de medição de ângulos possuem limites finitos e conhecidos. A região do plano que, garantidamente, contém a posição verdadeira do robô, não é determinada recorrendo à Teoria de Conjuntos, mas sim deduzida a partir de considerações de ordem geométrica que tomam em consideração as posições relativas do robô e das balizas e os limites dos erros de medição.

[4] Uma distribuição rectangular de probabilidade é a que caracteriza uma grandeza cujo valor pode ser, com igual probabilidade, qualquer um pertencente a um intervalo com limites finitos e conhecidos (Adams, 2002).

- caracterizar a incerteza de posição e também a incerteza de orientação, se se partir do princípio que os erros de medição têm limites finitos conhecidos;

- caracterizar a incerteza de posição devida aos erros aleatórios de medição, se se considerar que estes erros possuem distribuições de probabilidade gaussianas.

Em vez de recorrer à Teoria de Conjuntos, o novo método baseia-se nas propriedades geométricas da *superfície de incerteza de posição*, região do plano de navegação que é determinada a partir das medições de ângulos efectuadas num dado ponto do plano e que

- contém, garantidamente, a posição verdadeira do robô, se os erros de medição tiverem limites finitos conhecidos;

- contém, com um determinado *nível de confiança*[5], a posição verdadeira do robô, se os erros de medição possuírem distribuições gaussianas.

Quando é possível estabelecer valores máximos finitos para os erros de medição, as incertezas na posição e na orientação calculadas podem ser caracterizadas por um *erro máximo de posição* e um *erro máximo de orientação* (valores máximos do erro de posição e do erro de orientação, respectivamente).

Os dois algoritmos apresentados no ponto 5.6 destinam-se ao cálculo dos valores máximos que o erro de posição e o erro de orientação podem assumir num ponto do plano de navegação, para uma dada incerteza contida nas medidas dos ângulos. Estes algoritmos não requerem o cálculo de derivadas parciais com expressões analíticas difíceis de obter e não recorrem a aproximações. Mostrar-se-á que é possível usar o algoritmo de cálculo do erro máximo de posição para caracterizar a incerteza de posição devida aos erros aleatórios de medição com distribuições de probabilidade gaussianas.

No ponto 5.7 apresentam-se as conclusões deste capítulo e fazem-se algumas sugestões de trabalho futuro.

[5] O *nível de confiança* associado a um intervalo de valores que contém a medida de uma grandeza é a probabilidade de o valor verdadeiro dessa grandeza estar contido no intervalo (Clapham, 1996). Analogamente, o *nível de confiança* associado a uma determinada região do plano de navegação que contém a posição calculada do robô é a probabilidade de a posição verdadeira do robô se encontrar dentro dessa região.

5.1 Caracterização da Configuração de Balizas

As posições (conhecidas *a priori*) ocupadas no plano de navegação pelas três balizas requeridas para a autolocalização por triangulação determinam a figura geométrica por elas formada. Esta figura é um triângulo ou um segmento de recta que pode ser visto como um triângulo degenerado. Para caracterizar a família de triângulos semelhantes à qual pertence o triângulo formado pelas balizas e também o sentido em que estas se encontram ordenadas utilizam-se dois ângulos, σ e δ, definidos da seguinte maneira:

- *Seja σ um ângulo orientado tal que $-180^o < \sigma \leq 180^o$. O seu lado origem é o segmento de recta que une as balizas 1 e 3. O lado extremidade é a parte da recta que passa pelas balizas 1 e 2 cuja origem é a baliza 1 e não contém a baliza 2.*

- *Seja δ um ângulo orientado tal que $-180^o < \delta \leq 180^o$. O seu lado origem é o segmento de recta que une as balizas 2 e 3. O lado extremidade é o segmento de recta que une as balizas 1 e 2.*

Na Figura 5.1 e na Figura 5.2 indicam-se os intervalos a que pertencem σ e δ quando as balizas formam um triângulo e são ordenadas no sentido directo ou no sentido inverso, respectivamente. Na Figura 5.3 indicam-se os valores de σ e δ correspondentes às três possíveis ordenações de balizas colineares. Cada ordenação distingue-se das outras pelo número atribuído à baliza central.

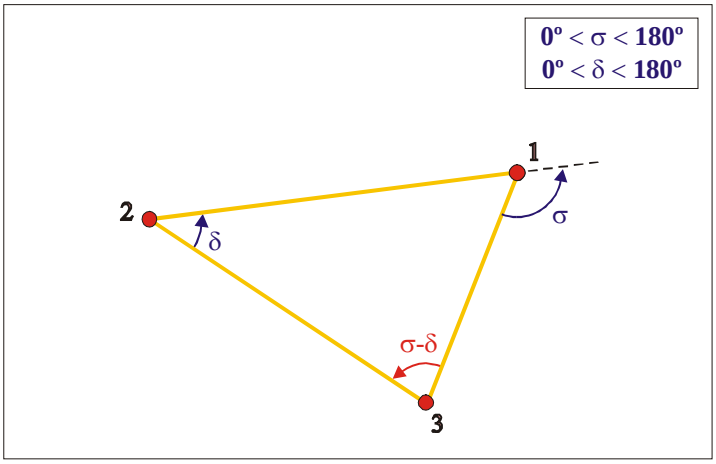


Figura 5.1: σ e δ com três balizas não colineares ordenadas no sentido directo.

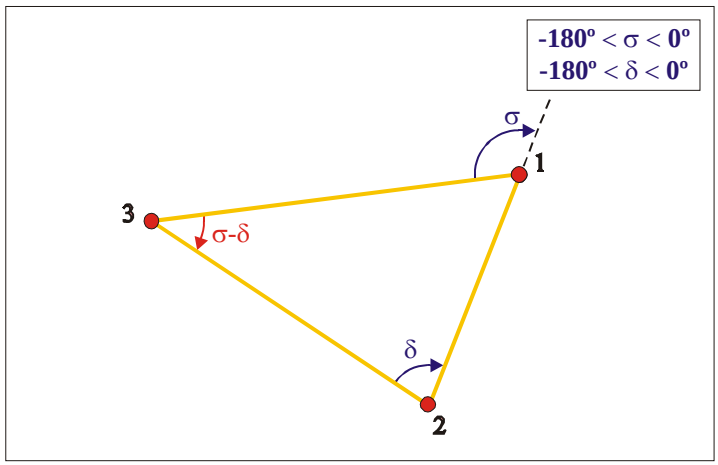


Figura 5.2: σ e δ com três balizas não colineares ordenadas no sentido inverso.

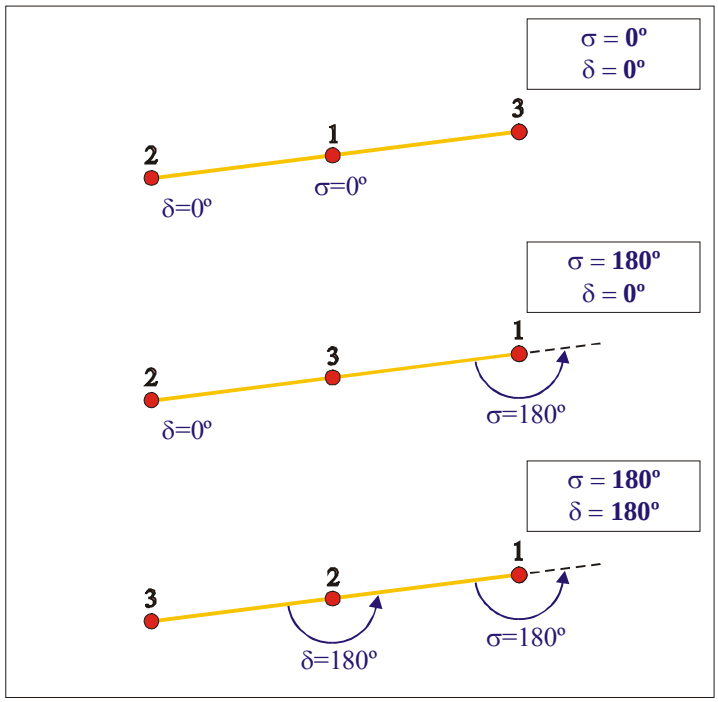


Figura 5.3: σ e δ com três balizas colineares.

O gráfico de δ em função de σ para todas as possíveis configurações de balizas encontra-se na Figura 5.4, que também exibe alguns casos particulares de configurações. A cada ponto no interior da superfície sombreada corresponde uma família de triângulos (de balizas) semelhantes. Nenhum ponto do seu exterior representa configurações válidas e, sobre o seu contorno, apenas os pontos G, H e I o fazem:

- O ponto G corresponde a todas as configurações de balizas colineares em que a baliza 1 se encontra no meio das balizas 2 e 3;

- O ponto H corresponde a todas as configurações de balizas colineares em que a baliza 3 se encontra no meio das balizas 1 e 2;

- O ponto I corresponde a todas as configurações de balizas colineares em que a baliza 2 se encontra no meio das balizas 1 e 3.

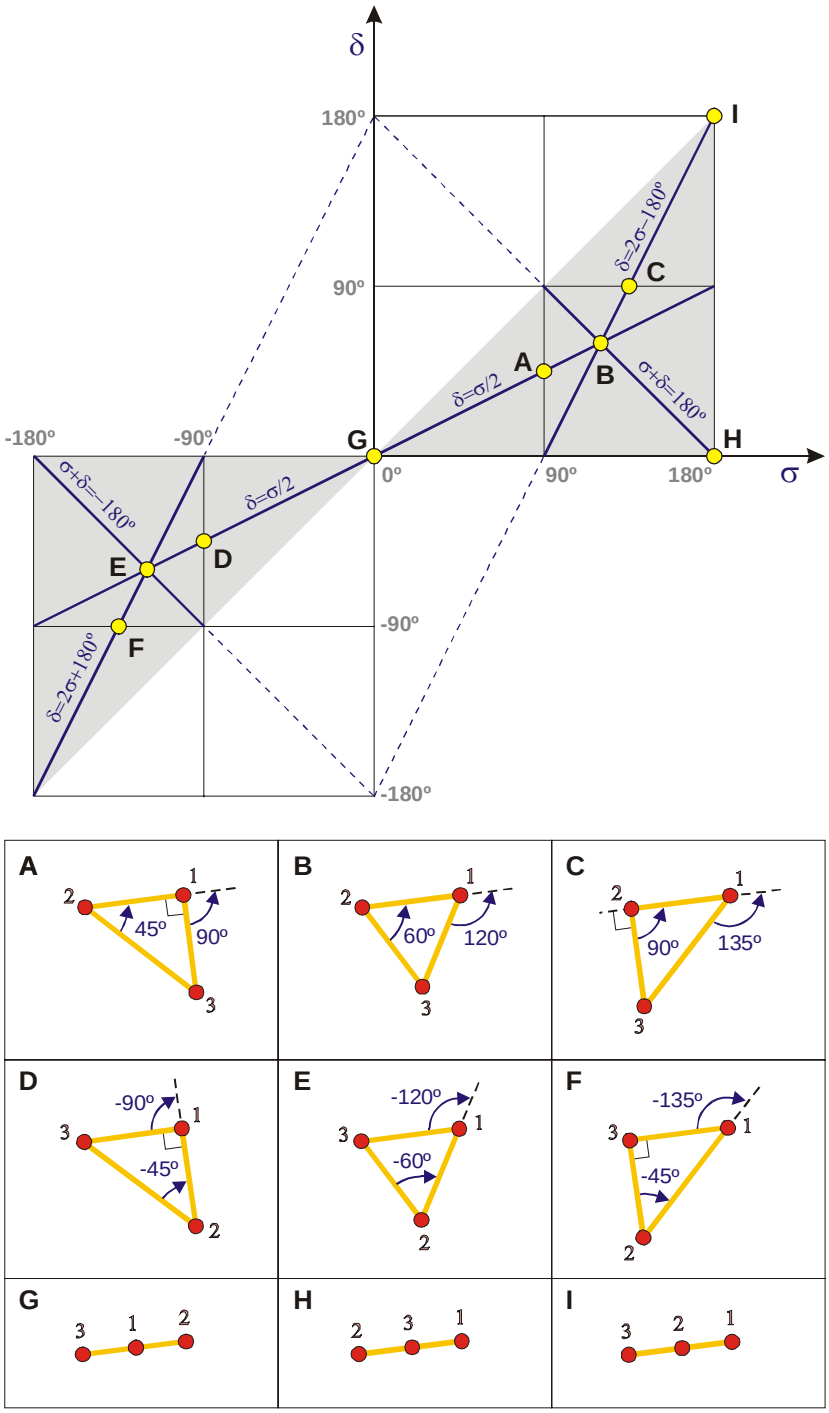


Figura 5.4: Gráfico de δ em função de σ para todas as possíveis configurações de balizas e alguns casos particulares. A cada ponto no interior da superfície sombreada corresponde uma família de triângulos (de balizas) semelhantes. Sobre o seu contorno, apenas os pontos G, H e I representam configurações válidas de balizas colineares. Não há nenhum ponto no exterior da superfície sombreada que represente configurações válidas.

A orientação do triângulo formado pelas balizas relativamente aos eixos em que se medem as coordenadas no plano de navegação fica determinada por um ângulo ϕ com a seguinte definição:

- *Seja ϕ um ângulo orientado tal que $-180^o < \phi \leq 180^o$. O seu lado origem é a imagem do semieixo positivo dos xx que resulta da translação associada ao vector cuja origem é a origem do referencial x0y e termina na baliza 1. O lado extremidade é a parte da recta que passa pelas balizas 1 e 2 cuja origem é a baliza 1 e não contém a baliza 2.*

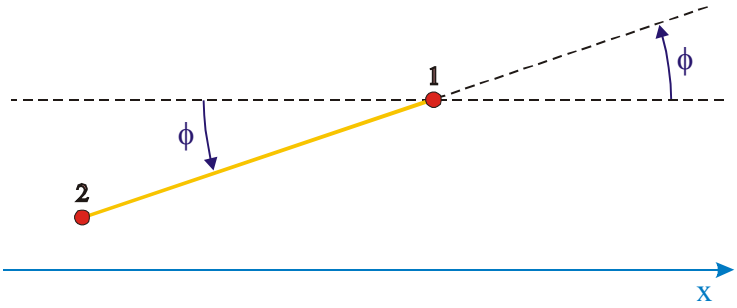


Figura 5.5: Definição de ϕ.

5.2 Definição dos Ângulos Formados pelos Segmentos de Recta que Unem o Robô a Cada Baliza

Kuipers e Levitt (1988) descrevem um método de autolocalização baseado na ordem com que as balizas são vistas por um observador (ver também Sutherland, 1994; Thompson *et al.,* 1996). Na situação da Figura 5.6, a baliza 1 é vista à direita da baliza 2 por um observador que se encontre dentro da zona do plano de navegação sombreada a cinzento mais claro. Quando o observador se desloca para a zona sombreada a cinzento mais escuro, é a baliza 2 que passa a ser vista à direita da baliza 1. Assim, a ordem pela qual as balizas são vistas pode ser utilizada para identificar em qual das duas zonas se encontra o observador. Aplicando o mesmo raciocínio a três balizas não colineares obtêm-se sete zonas diferentes. No entanto, a autolocalização baseada exclusivamente na ordem pela qual as balizas são vistas é bastante limitada pois, apesar de saber em que zona está, o observador não consegue determinar em que ponto dessa zona se encontra.

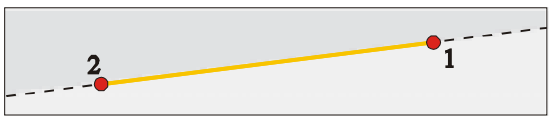


Figura 5.6: Divisão do plano de navegação em duas zonas, de acordo com a ordem pela qual as balizas são vistas por um observador. Na zona sombreada a cinzento mais claro, a baliza 1 é vista à direita da baliza 2. Na zona sombreada a cinzento mais escuro, é a baliza 2 que é vista à direita da baliza 1.

Como seguidamente se demonstra, a utilização de ângulos orientados possibilita a divisão do plano de navegação nas mesmas zonas. Mas vai mais longe, pois permite também que o observador determine sem ambiguidade as coordenadas do ponto em que se encontra.

Seja α o menor dos ângulos formados pelos segmentos de recta que unem o robô a duas balizas não colineares com o robô[6]. A medida deste ângulo determina dois arcos de circunferência[7] sobre os quais o robô se pode encontrar, no plano de navegação (Kuipers e Levitt, 1988; Sutherland e Thompson, 1993; Sutherland, 1994; Araújo, 1999), como se pode ver na Figura 5.7. No entanto, se os ângulos forem orientados e um desses segmentos de recta for sempre o lado origem, apenas a um destes arcos corresponde um ângulo de valor α. A análise da Figura 5.7 permite concluir facilmente que ao outro arco corresponde um valor 360º-α. Desta forma, a cada arco corresponde apenas um ângulo bem definido. Isto é também válido nos casos particulares em que o robô e as balizas são colineares[7].

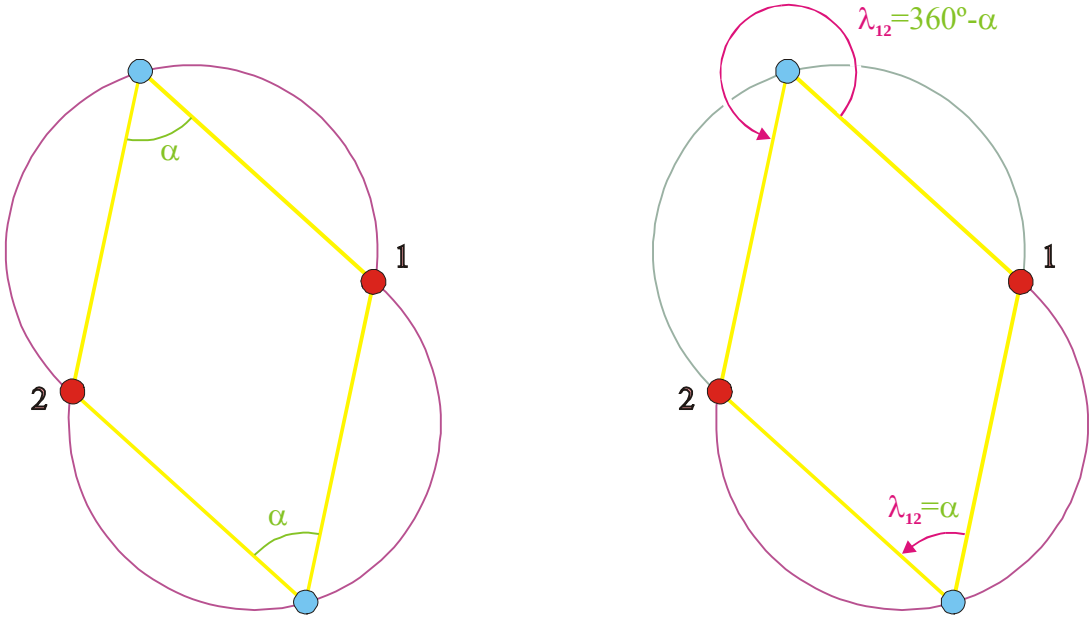


Figura 5.7: Utilização de ângulos orientados para reduzir a ambiguidade de posição.

[6] Isto implica $\alpha<180º$.

[7] Se o robô e as duas balizas forem colineares:

- se o robô se encontrar entre as duas balizas, então $\alpha=180º$ e os dois arcos degeneram no segmento de recta que une as duas balizas, que pode ser visto como um único arco de circunferência de raio infinito;
- se o robô não se encontrar entre as duas balizas, então $\alpha=0º$ e os dois arcos degeneram em duas semi-rectas com origem nas balizas, direcção do segmento de recta, que une as balizas e não contêm esse segmento. As semi-rectas podem ser vistas como um único arco de circunferência de raio infinito.

Recorrendo a três balizas é possível obter dois arcos que, em geral, só se intersectam numa das balizas e na posição em que se encontra o robô[8]. O conhecimento dos dois ângulos orientados correspondentes aos arcos permite determinar sem ambiguidade a posição do robô, desde que este se encontre fora da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares.

De acordo com o exposto, para eliminar a ambiguidade na determinação da posição de um robô que se encontra fora da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares sugerem-se as seguintes definições dos ângulos λ_{12}, λ_{23} e λ_{31} (Figura 5.8):

- *Seja λ_{12} um ângulo orientado tal que $0^o \leq \lambda_{12} < 360^o$. O seu lado origem é o segmento de recta que une o robô à baliza 1 e o seu lado extremidade é o segmento de recta que une o robô à baliza 2;*

- *Seja λ_{23} um ângulo orientado tal que $0^o \leq \lambda_{23} < 360^o$. O seu lado origem é o segmento de recta que une o robô à baliza 2 e o seu lado extremidade é o segmento de recta que une o robô à baliza 3;*

- *Seja λ_{31} um ângulo orientado tal que $0^o \leq \lambda_{31} < 360^o$. O seu lado origem é o segmento de recta que une o robô à baliza 3 e o seu lado extremidade é o segmento de recta que une o robô à baliza 1.*

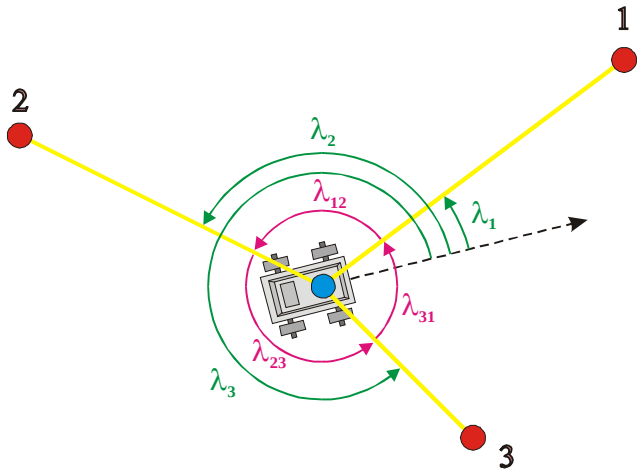


Figura 5.8: Definição dos ângulos λ_{12}, λ_{23}, λ_{31}, λ_1, λ_2 e λ_3.

[8] Dois arcos de circunferência com um ponto comum (uma das balizas) só se podem intersectar em mais um ponto, a menos que estejam sobrepostos.

Quando algum dos ângulos λ_{12}, λ_{23} ou λ_{31} é igual a 0º, uma das balizas encontra-se sobre o segmento de recta que une o robô a outra baliza. Há dois motivos que podem impedir a autolocalização por triangulação nessas circunstâncias (parte-se do princípio que há apenas três balizas disponíveis):

1. Há ângulos que não podem ser medidos porque a baliza mais afastada do robô é ocultada pela mais próxima[9] ou então porque o instrumento de medição de ângulos não tem a capacidade de detectar simultaneamente mais do que uma baliza[10];

2. As três balizas são detectadas pelo instrumento de medição de ângulos mas o algoritmo de triangulação não funciona se algum dos ângulos λ_{12}, λ_{23} ou λ_{31} for nulo ou valer 360º [11];

Os ângulos λ_{12}, λ_{23} e λ_{31} podem ser medidos directamente. Mas também podem ser calculados a partir das medidas dos ângulos λ_1, λ_2 e λ_3, definidos da seguinte maneira:

- *Seja λ_1 um ângulo orientado tal que $0º \leq \lambda_1 < 360º$. O seu lado origem é um semieixo de referência fixo no robô e o seu lado extremidade é o segmento de recta que une o robô à baliza 1;*

- *Seja λ_2 um ângulo orientado tal que $0º \leq \lambda_2 < 360º$. O seu lado origem é um semieixo de referência fixo no robô e o seu lado extremidade é o segmento de recta que une o robô à baliza 2;*

- *Seja λ_3 um ângulo orientado tal que $0º \leq \lambda_3 < 360º$. O seu lado origem é um semieixo de referência fixo no robô e o seu lado extremidade é o segmento de recta que une o robô à baliza 3.*

[9] Estas considerações são feitas para balizas pontuais. Com balizas de dimensões não desprezáveis pode acontecer que a baliza mais afastada só seja detectada pelo instrumento de medição de ângulos quando o ângulo formado pelos segmentos de recta que unem o robô às balizas atinge um determinado valor acima de 0º ou abaixo de 360º.

[10] Na prática, com tal instrumento, o ângulo formado pelos segmentos de recta que unem o robô às balizas só é medido quando atinge um determinado valor acima de 0º ou abaixo de 360º.

[11] Devido aos erros cometidos na medição dos ângulos, pode acontecer que algum dos ângulos λ_{12}, λ_{23} ou λ_{31} seja interpretado como 0º ou 360º quando na verdade é apenas próximo de um desses valores.

De acordo com o valor de λ_{12} é possível dividir o plano de navegação em duas zonas (Figura 5.9). Numa dessas zonas λ_{12} é sempre superior a 180º. Na outra zona λ_{12} é sempre inferior a 180º. Ao longo fronteira entre as duas zonas, λ_{12} é igual 180º sobre o segmento que une as duas balizas e igual a 0º fora desse segmento. Pode aplicar-se aos ângulos λ_{23} e λ_{31} um análise semelhante. O resultado é a divisão do plano de navegação nas zonas apresentadas na Figura 5.10 para balizas não colineares ordenadas no sentido directo, na Figura 5.11 para balizas não colineares ordenadas no sentido inverso e na Figura 5.12 para os três tipos de configurações de balizas colineares. Em qualquer ponto do plano de navegação verifica-se sempre que $\lambda_{12}+\lambda_{23}+\lambda_{31}=360º$ ou então $\lambda_{12}+\lambda_{23}+\lambda_{31}=720º$.

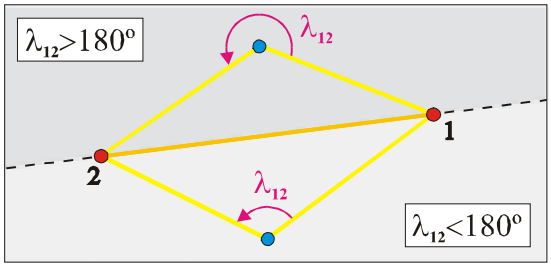


Figura 5.9: Divisão do plano de navegação em duas zonas, de acordo com o valor de λ_{12}.

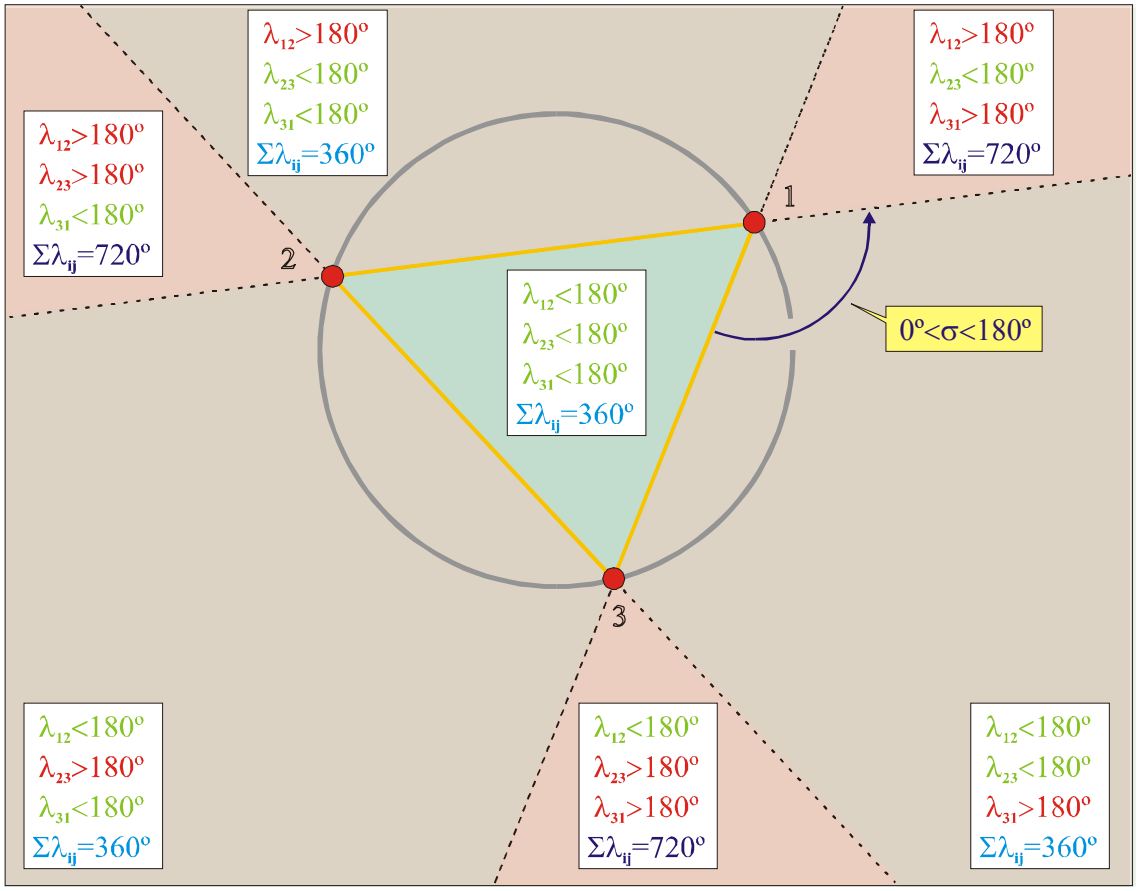


Figura 5.10: Divisão do plano de navegação em sete zonas, de acordo com os valores de λ_{12}, λ_{23} e λ_{31}, para balizas não colineares ordenadas no sentido directo.

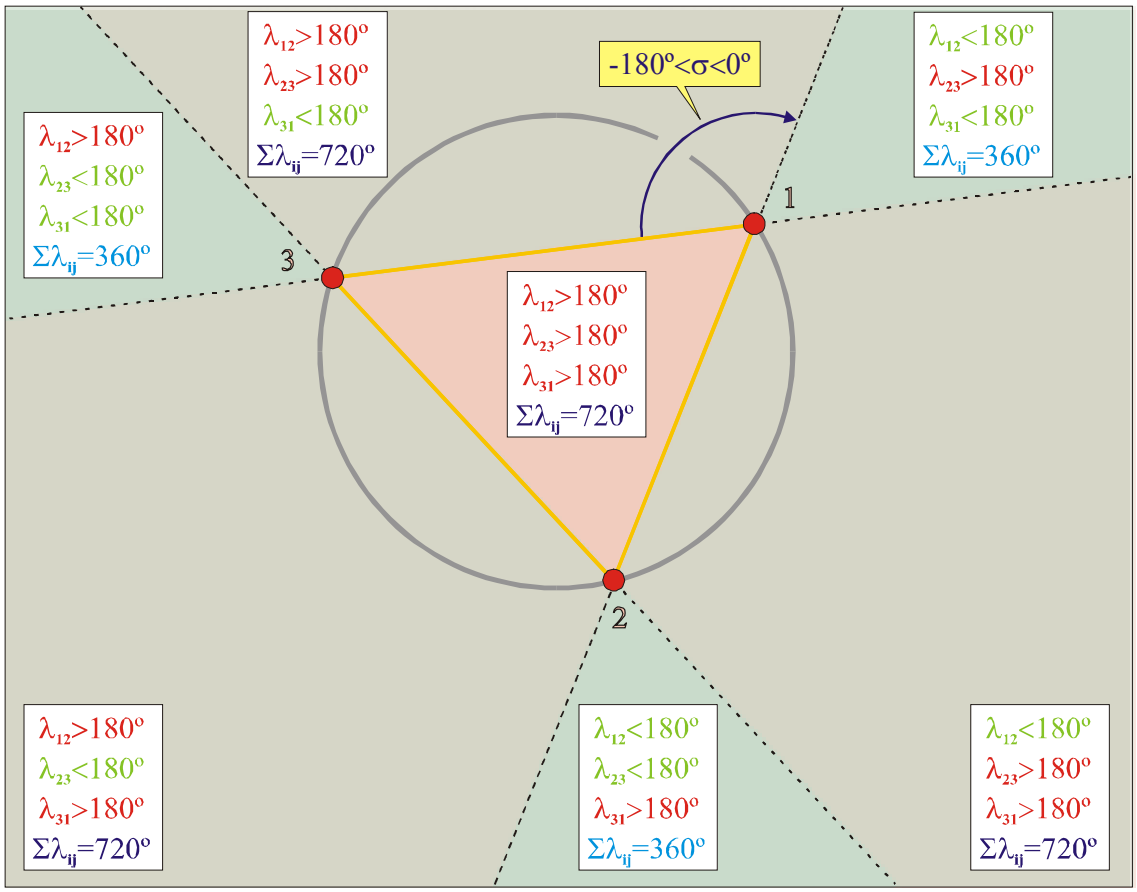


Figura 5.11: Divisão do plano de navegação em sete zonas, de acordo com os valores de λ_{12}, λ_{23} e λ_{31}, para balizas não colineares ordenadas no sentido inverso.

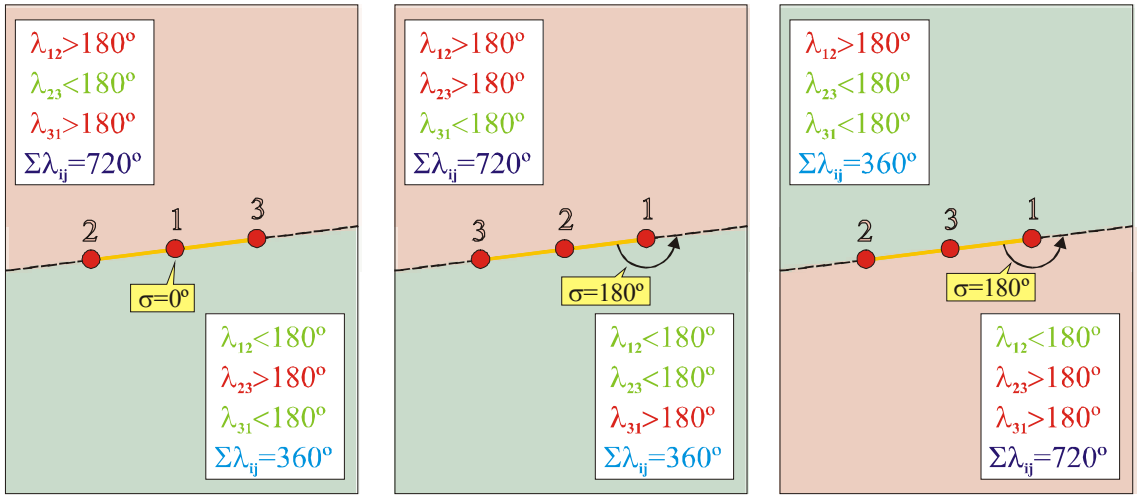


Figura 5.12: Divisão do plano de navegação em duas zonas, de acordo com os valores de λ_{12}, λ_{23} e λ_{31}, para os três tipos de configurações com balizas colineares.

Um dos três ângulos λ_{12}, λ_{23} e λ_{31} é redundante na determinação da posição do robô, que pode ser calculada a partir de apenas dois. Doravante utilizar-se-ão λ_{12} e λ_{31}. De acordo com os valores de λ_{12} e λ_{31} é possível dividir o plano de navegação nas zonas apresentadas na Figura 5.13, para cada um dos cinco tipos de configuração de balizas.

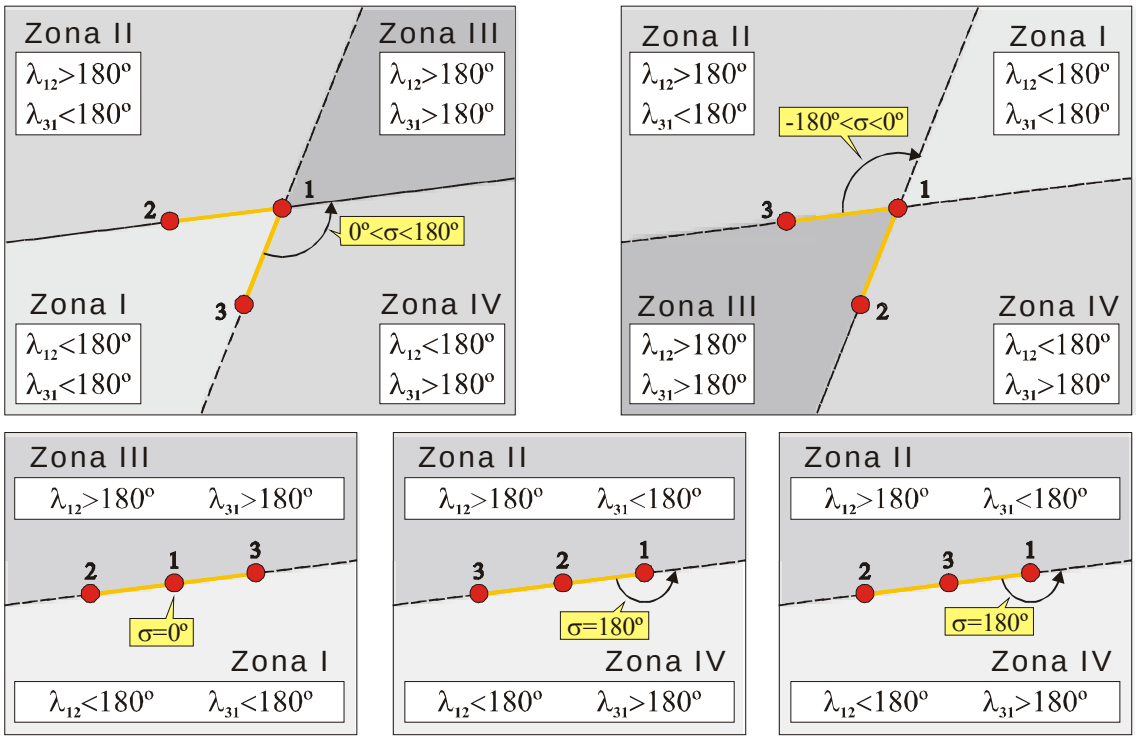


Figura 5.13: Divisão do plano de navegação, de acordo com os valores de λ_{12} e λ_{31}.

Uma circunferência que passa por duas balizas pode ser decomposta em dois arcos cujas extremidades são essas balizas. Se a um desses arcos corresponder um ângulo λ_{12} de valor α, então ao outro arco corresponde[12] um ângulo λ_{12} de valor $180^o+\alpha$, como se vê na Figura 5.14. O mesmo princípio é aplicável a λ_{31}. Assim, na Figura 5.15 e na Figura 5.16 podem ver-se os valores particulares que os ângulos λ_{12} e λ_{31} assumem em cada um dos três arcos em que se pode dividir a circunferência definida por três balizas não colineares, ordenadas no sentido directo ou no sentido inverso. Esses valores, como só dependem de σ e δ, são determinados exclusivamente pela configuração de balizas e, portanto, conhecidos *a priori*. Isto é muito útil, pois constitui um meio de o algoritmo de triangulação detectar a presença do robô sobre a circunferência, podendo assim desencadear uma acção adequada a esta situação na qual a autolocalização não é possível. Quando as três balizas são colineares, a autolocalização não é possível sobre a recta que passa pelas três balizas[13]. A presença do robô sobre essa recta é detectada pelo facto λ_{12} e λ_{31} assumirem valores 0º ou 180º.

[12] Num quadrilátero inscrito numa circunferência, a soma de dois ângulos internos cujos vértices são vértices opostos do quadrilátero é igual a 180º (Clapham, 1996).

[13] Esta recta pode ser vista como uma degeneração da circunferência que passa por três balizas não colineares.

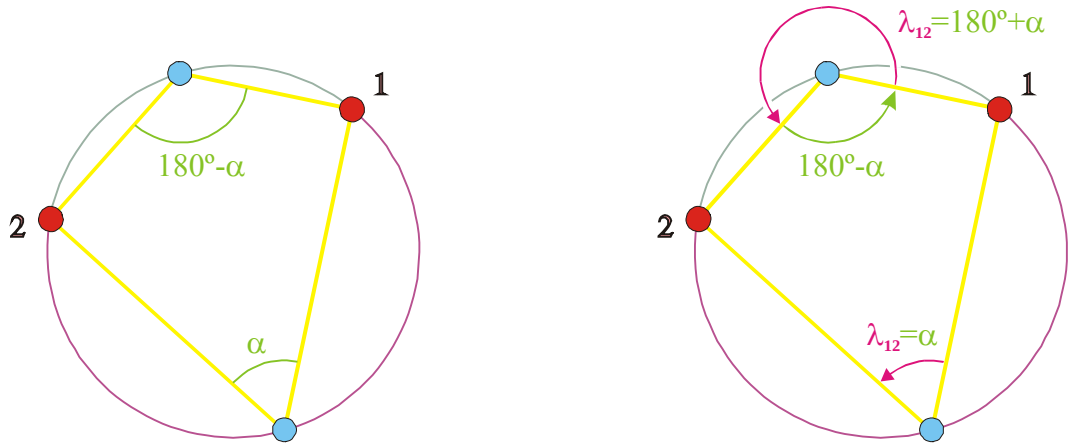


Figura 5.14: Relação entre os valores de λ_{12} correspondentes aos dois arcos de circunferência cujas extremidades são as balizas 1 e 2 e pertencem a uma mesma circunferência.

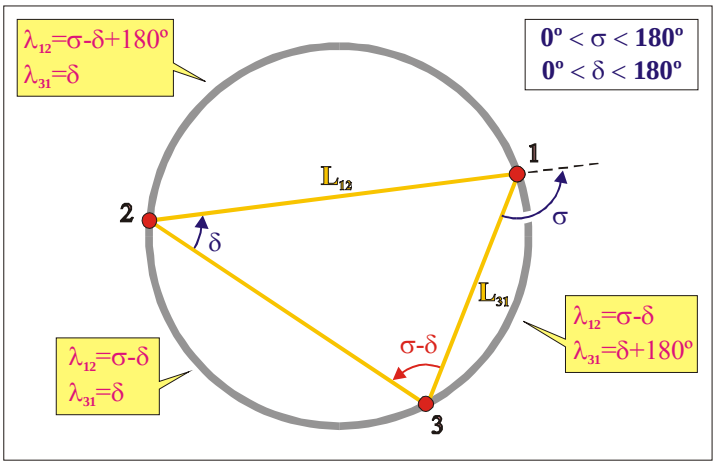


Figura 5.15: Valores particulares de λ_{12} e λ_{31} sobre a circunferência que passa pelas três balizas ordenadas no sentido directo.

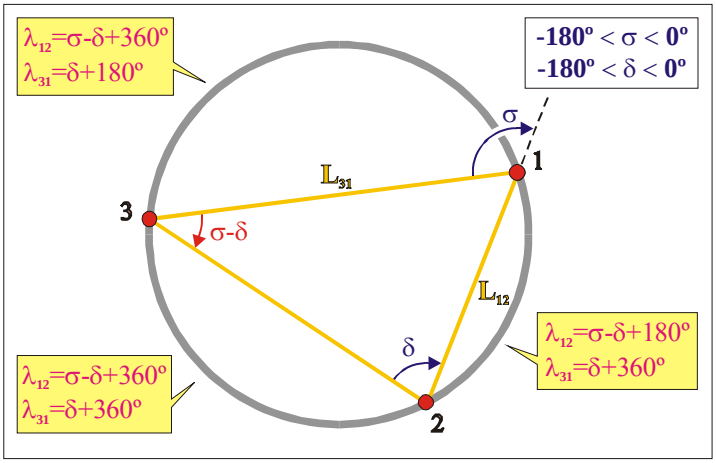


Figura 5.16: Valores particulares de λ_{12} e λ_{31} sobre a circunferência que passa pelas três balizas ordenadas no sentido inverso.

5.3 Definição da Orientação do Robô

A orientação do robô é o ângulo θ_R para o qual se sugere a seguinte definição (Figura 5.17):

- *Seja θ_R um ângulo orientado tal que $-180^o < \theta_R \leq 180^o$. O seu lado origem é o semieixo positivo dos xx do referencial x0y definido no plano de navegação. O lado extremidade é um semieixo de referência fixo no robô.*

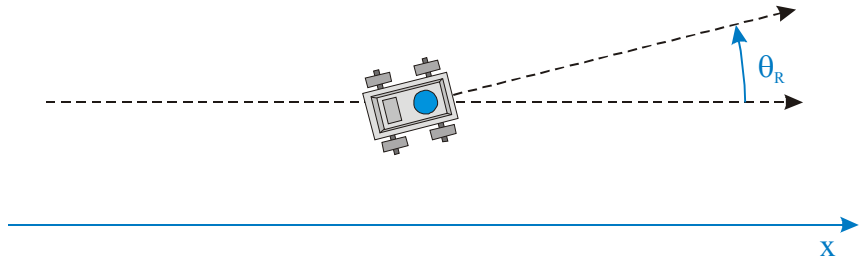


Figura 5.17: Definição de θ_R.

Para determinar θ_R é útil considerar o ângulo τ definido da seguinte maneira (Figura 5.18):

- *Seja τ um ângulo orientado tal que $-180^o < \tau \leq 180^o$. O seu lado origem é o segmento de recta que une as balizas 1 e 2. O lado extremidade é o segmento de recta que une o robô e a baliza 1.*

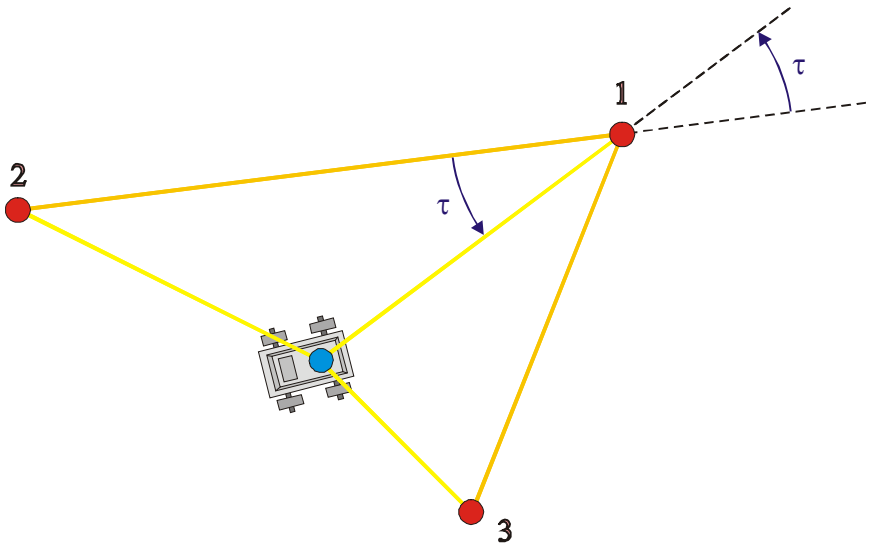


Figura 5.18: Definição de τ.

O valor de τ em cada ponto fica determinado pelos valores de λ_{12} e λ_{31} que ocorrem nesse ponto[14]. O calculo de θ_R implica conhecer também um dos ângulos λ_1, λ_2 ou λ_3 (utilizar-se-á λ_1) e pode ser feito recorrendo ao seguinte algoritmo (Figura 5.19):

1. $\theta_R = \phi + \tau - \lambda_1$
2. Se $\theta_R \leq -180^o$ então $\theta_R = \theta_R + 360^o$
3. Se $\theta_R > 180^o$ então $\theta_R = \theta_R - 360^o$

As linhas 2 e 3 destinam-se a garantir que $-180^o < \theta_R \leq 180^o$, de acordo com a definição de θ_R.

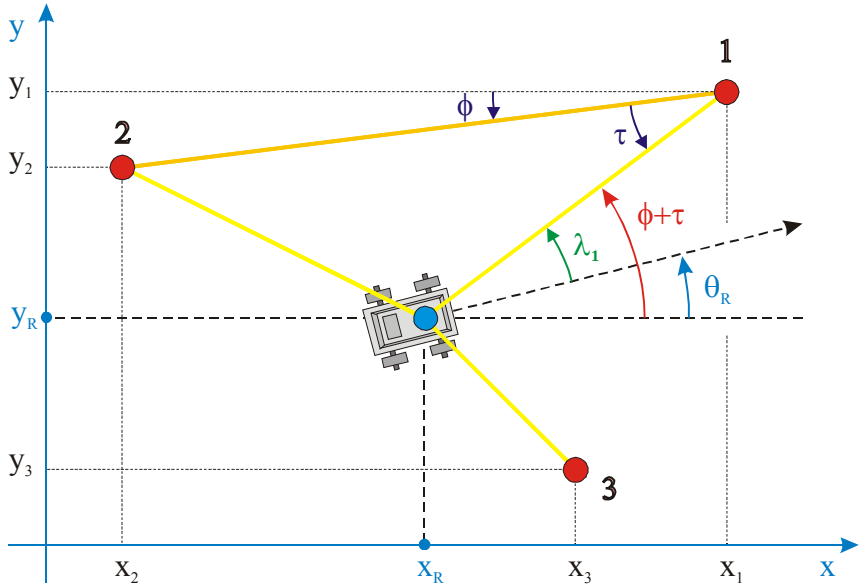


Figura 5.19: Cálculo de θ_R.

Há uma relação entre τ e λ_{12}, como se pode verificar na Figura 5.20.

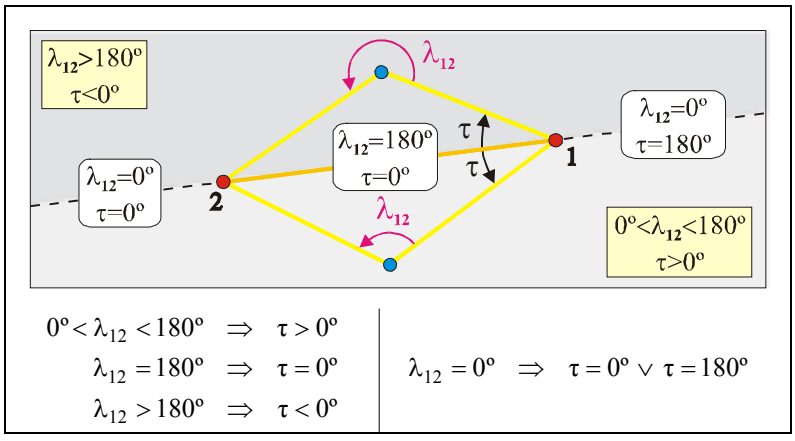


Figura 5.20: Relação entre τ e λ_{12}.

[14] A especificidade de cada algoritmo de triangulação, no que se refere à determinação da orientação do robô, está no modo de calcular τ.

5.4 Nova Especificação do Problema da Autolocalização por Triangulação

De acordo com as definições de ângulos propostas nos pontos anteriores, sugere-se uma nova especificação do problema da autolocalização absoluta a duas dimensões por triangulação (Figura 5.21): *dadas três balizas distinguíveis situadas em posições conhecidas de um plano de navegação no qual se definiu um referencial ortonormado x0y e também os valores assumidos, num dado instante, pelos ângulos λ_1, λ_{12} e λ_{31} (definidos do modo descrito no ponto 5.2), determinar sem recorrer a suposições sobre movimentos anteriores:*

- *a posição do robô, ou seja, as suas coordenadas x_R e y_R no referencial x0y, no instante considerado;*

- *a orientação do robô, ou seja, o ângulo θ_R, no instante considerado.*

  θ_R tem a seguinte definição:

  - *Seja θ_R um ângulo orientado tal que $-180^o < \theta_R \leq 180^o$. O seu lado origem é o semieixo positivo dos xx do referencial x0y definido no plano de navegação. O lado extremidade é um semieixo de referência fixo no robô.*

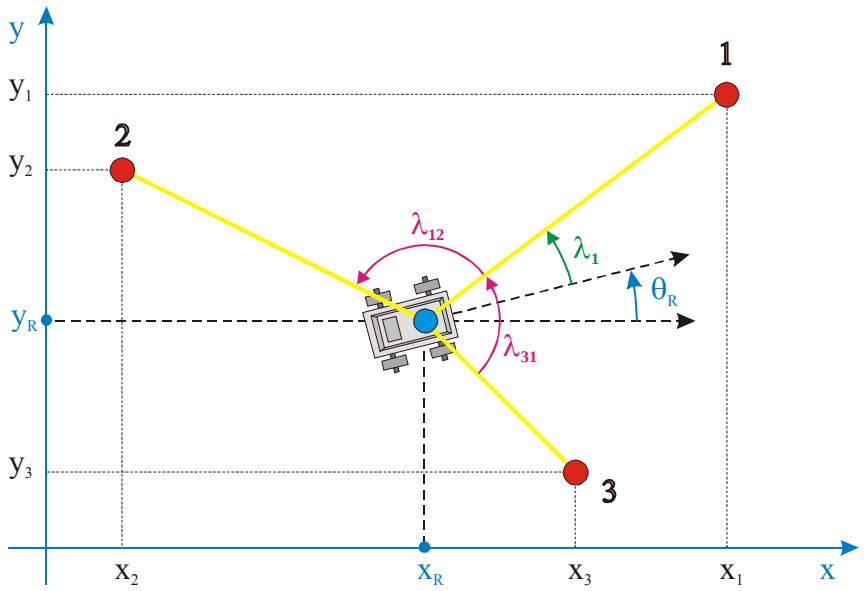


Figura 5.21: Autolocalização por triangulação.

5.5 Relação Entre a Posição do Robô e os Ângulos λ_{12} e λ_{31}

Os ângulos λ_{12} e λ_{31} são usados como variáveis de entrada[15] em algoritmos de cálculo de posição por triangulação, pelo que é pertinente representar, num referencial ortonormado $\lambda_{12}0\lambda_{31}$, os pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação, para uma dada configuração de balizas. É o que se faz na Figura 5.22, para balizas não colineares ordenadas no sentido directo e na Figura 5.23, para balizas não colineares ordenadas no sentido inverso. Os três casos correspondentes a balizas colineares estão representados na Figura 5.24, na Figura 5.25e na Figura 5.26. Nos parágrafos que se seguem far-se-á referência apenas a algumas das principais características dos gráficos apresentados. Estes possuem propriedades geométricas interessantes que merecem uma análise mais detalhada do que a que é oportuna efectuar neste trabalho.

No referencial $\lambda_{12}0\lambda_{31}$ verifica-se que às três balizas correspondem rectas. Se as balizas forem colineares, estas rectas formam dois triângulos rectângulos isósceles iguais, cada um deles com dois lados paralelos aos eixos coordenados, dentro dos quais se encontram os pontos que têm imagem no plano de navegação[16]. Os dois lados iguais de cada triângulo medem sempre 180º e as posições ocupadas pelos polígonos dependem exclusivamente do número de ordem da baliza central (Figura 5.24, Figura 5.25e Figura 5.26).

Quando as balizas são não colineares (Figura 5.22 e Figura 5.23), um dos triângulos descritos no parágrafo anterior divide-se em três superfícies. A soma das áreas das superfícies cujos pontos têm imagem no plano de navegação permanece constante e igual ao dobro da área de um dos triângulos.

[15] λ_{12} e λ_{31} podem ser medidos directamente ou então calculados a partir das medidas de λ_1 λ_2 e λ_3. Neste segundo caso, seria mais rigoroso considerar λ_1 λ_2 e λ_3 as variáveis de entrada do algoritmo de determinação de posição utilizado. No entanto, a análise do problema da determinação de posição por triangulação pode, em ambos os casos, ser feita recorrendo a λ_{12} e λ_{31}.

[16] Na Figura 5.22 e seguintes, até à Figura 5.26, as superfícies cujos pontos têm imagem no plano de navegação aparecem sombreadas a amarelo. As superfícies formadas por pontos do referencial $\lambda_{12}0\lambda_{31}$ que não têm imagem no plano de navegação estão representadas a branco.

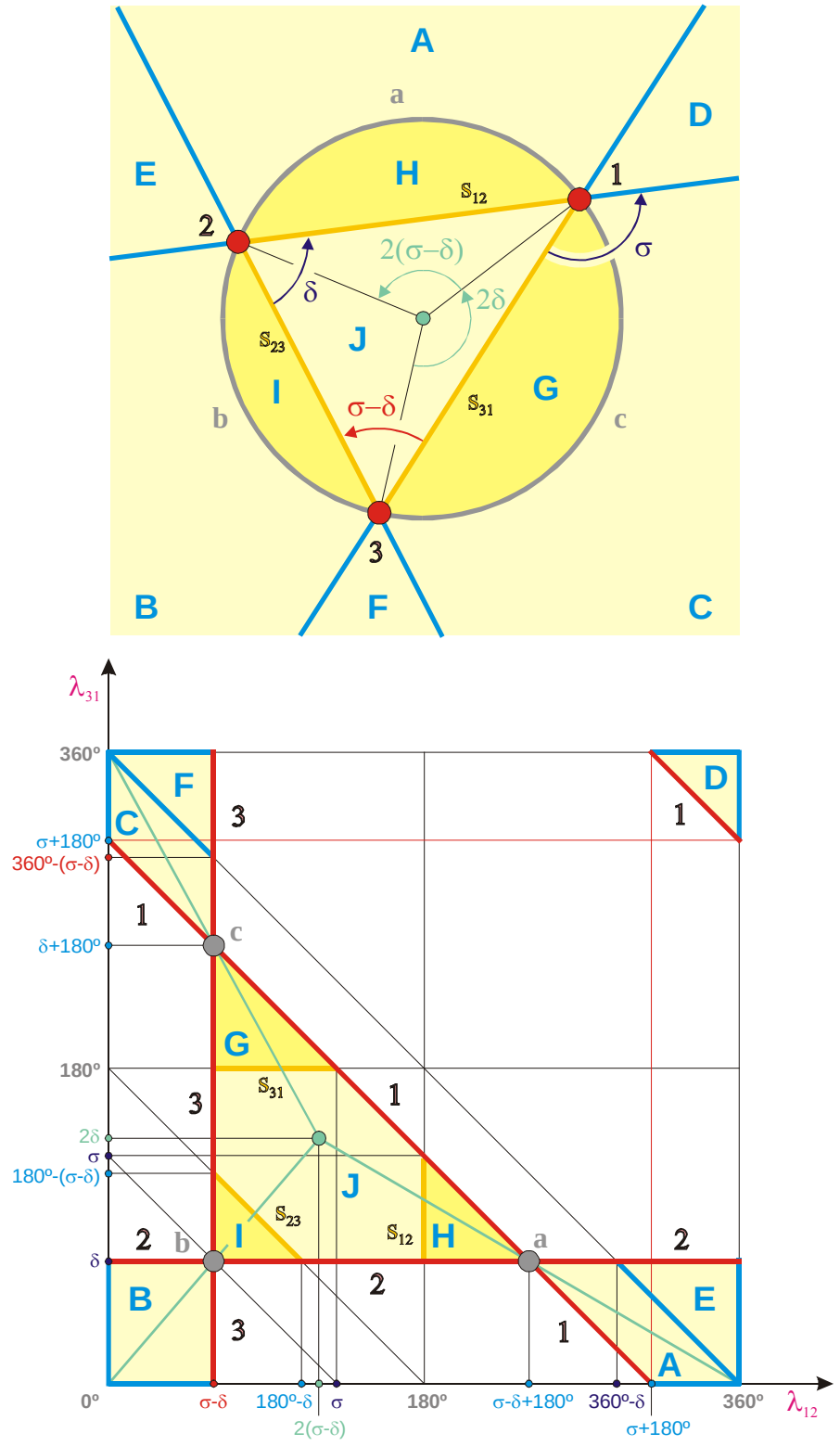


Figura 5.22: Representação no referencial $\lambda_{12}0\lambda_{31}$ dos pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação quando as três balizas são não colineares e estão ordenadas no sentido directo.

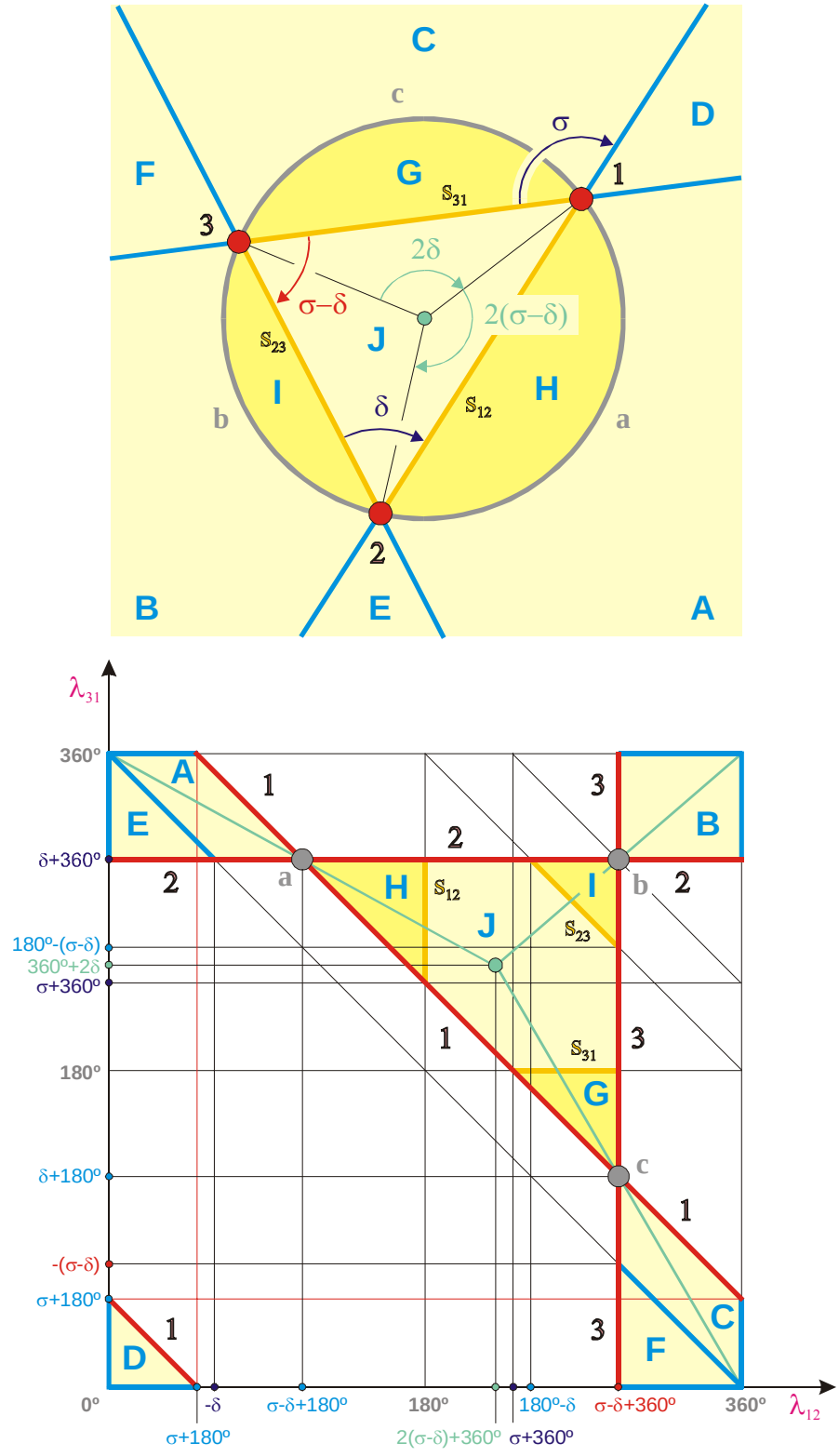


Figura 5.23: Representação no referencial $\lambda_{12}0\lambda_{31}$ dos pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação quando as três balizas são não colineares e estão ordenadas no sentido inverso.

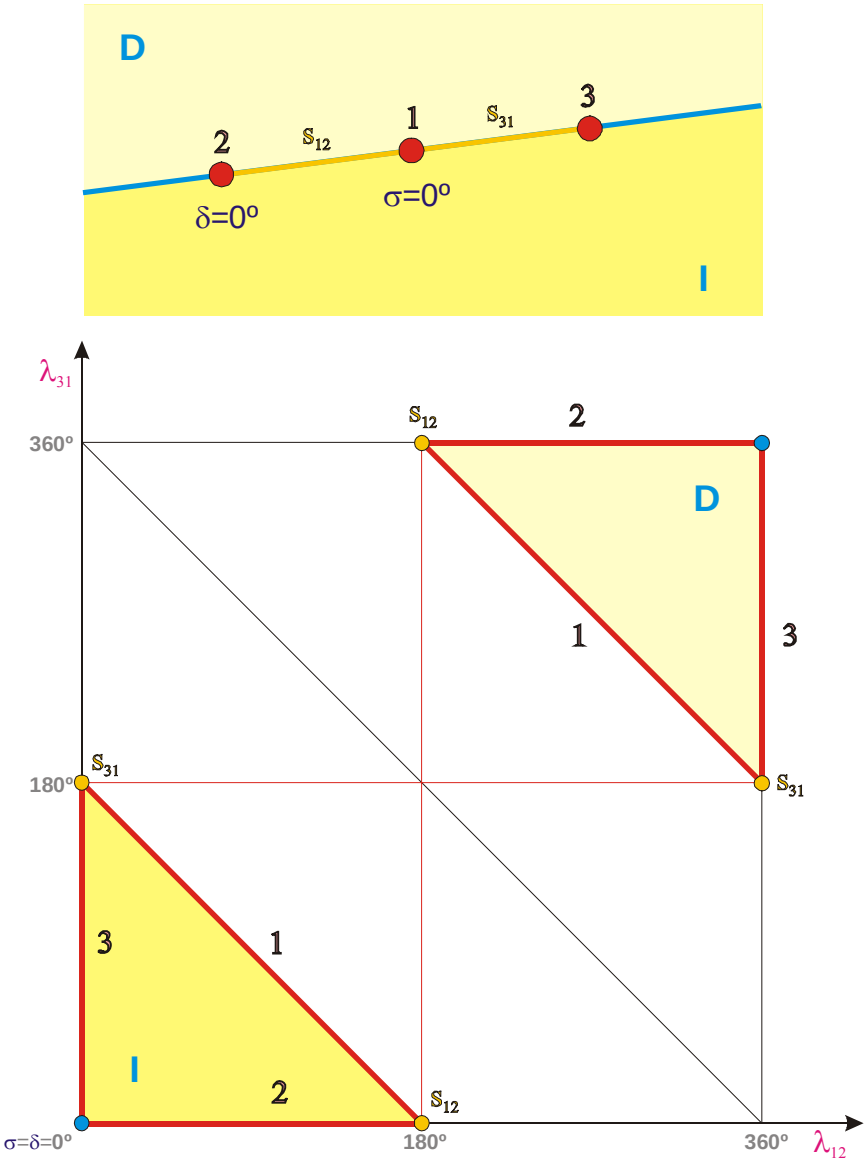


Figura 5.24: Representação no referencial $\lambda_{12}0\lambda_{31}$ dos pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação quando a baliza 1 se encontra sobre o segmento de recta que une as balizas 2 e 3.

Os vértices do triângulo subsistente correspondem, no plano de navegação, aos três arcos em que se divide a circunferência que passa pelas três balizas. A superfície no interior do triângulo corresponde, no plano de navegação, ao círculo delimitado por essa circunferência. A posição que o triângulo ocupa no referencial $\lambda_{12}0\lambda_{31}$ depende da família de triângulos semelhantes à qual pertence o triângulo definido pelas três balizas e também do sentido em que estas são ordenadas. Os ângulos σ e δ podem calcular-se a partir das coordenadas do ponto em que se intersectam as rectas correspondentes às balizas 2 e 3.

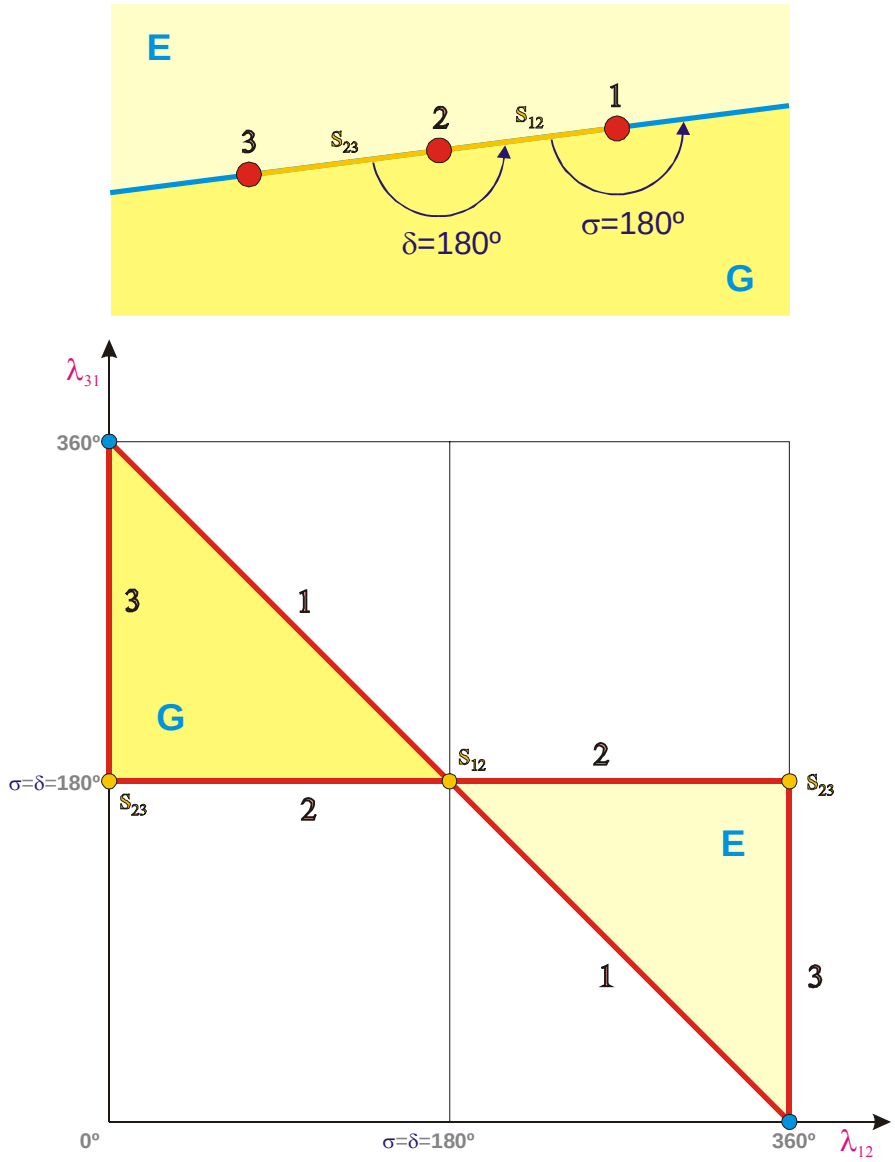


Figura 5.25: Representação no referencial $\lambda_{12}0\lambda_{31}$ dos pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação quando a baliza 2 se encontra sobre o segmento de recta que une as balizas 1 e 3.

No referencial $\lambda_{12}0\lambda_{31}$, o triângulo delimitado pelas rectas correspondentes a três balizas ordenadas no sentido directo pode ocupar qualquer posição dentro do triângulo cujos vértices são os pontos (0º, 0º), (360º, 0º) e (0º, 360º) (Figura 5.27). O triângulo delimitado pelas rectas correspondentes a três balizas ordenadas no sentido inverso pode ocupar qualquer posição dentro do triângulo cujos vértices são os pontos (360º, 0º), (0º, 360º) e (360º, 360º) (Figura 5.28).

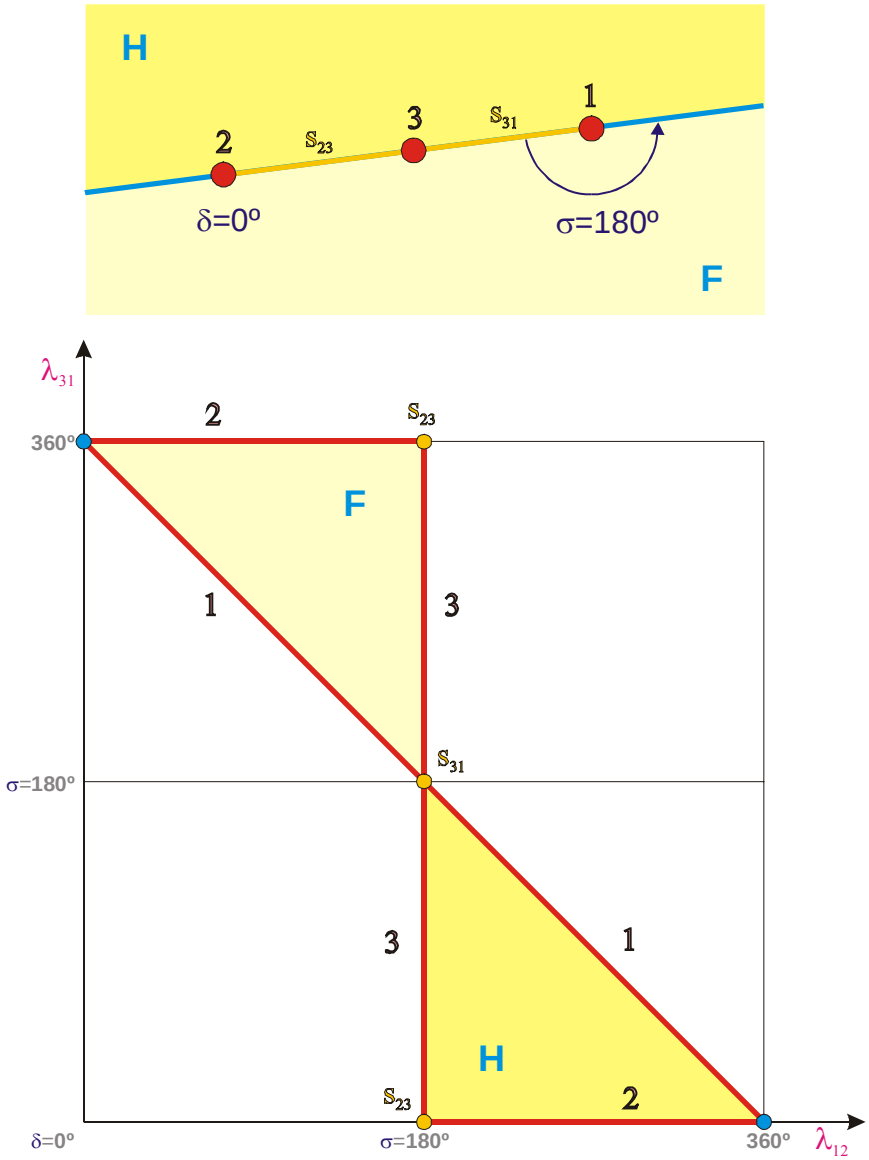


Figura 5.26: Representação no referencial $\lambda_{12}0\lambda_{31}$ dos pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação quando a baliza 3 se encontra sobre o segmento de recta que une as balizas 1 e 2.

Ao triângulo formado, no plano de navegação, por três balizas não colineares corresponde, no referencial $\lambda_{12}0\lambda_{31}$, um hexágono que resulta da intersecção do triângulo delimitado pelas rectas correspondentes às balizas com um outro, também rectângulo isósceles e com os lados iguais medindo sempre 180º. Se, no plano de navegação, as balizas estiverem ordenadas no sentido directo, os vértices deste segundo triângulo são os pontos (180º, 0º), (0º, 180º) e (180º, 180º) (Figura 5.27). Os vértices do triângulo passam a ser os pontos (360º, 180º), (180º, 360º) e (180º, 180º) se as balizas estiverem ordenadas no sentido inverso (Figura 5.28).

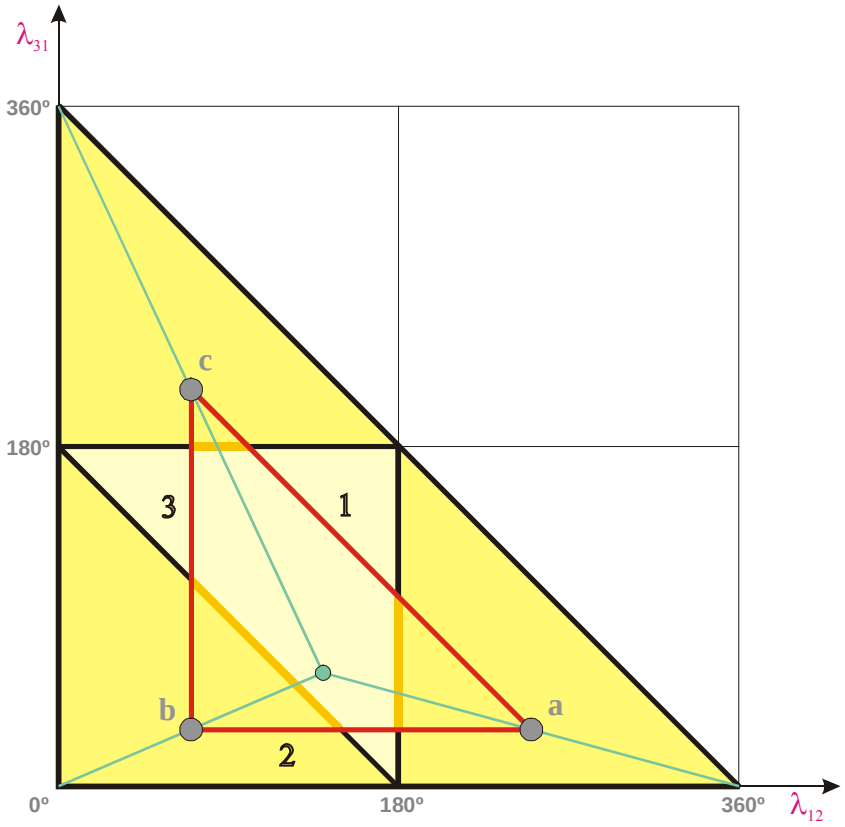


Figura 5.27: Ao triângulo formado pelas três balizas no plano de navegação corresponde um hexágono (que, no limite, pode ser degenerado) contido num triângulo no plano λ_{12}, λ_{31}.

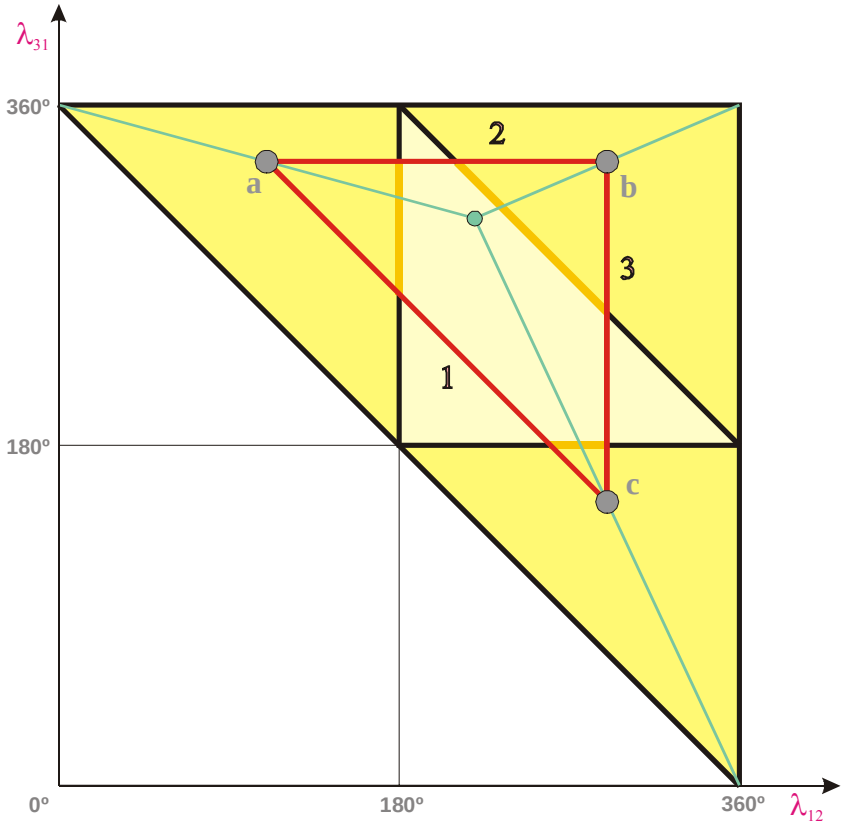


Figura 5.28: Ao triângulo formado pelas três balizas no plano de navegação corresponde um hexágono (que, no limite, pode ser degenerado) contido num triângulo no plano λ_{12}, λ_{31}.

5.6 Análise das Incertezas de Posição e de Orientação

A *posição calculada* mediante um determinado algoritmo geralmente não coincide com a *posição verdadeira*[17] do robô e a *orientação calculada* também é diferente da sua *orientação verdadeira*. Existem um *erro de posição* e um *erro de orientação* (Figura 5.29) que aqui se definem da seguinte maneira:

- O *erro de posição* ΔP_R é a distância entre a *posição calculada* P_{Rcalc} e a *posição verdadeira* P_R. Corresponde ao raio da circunferência que está centrada na posição calculada e contém a posição verdadeira (Figura 5.30).

- O *erro de orientação* $\Delta\theta_R$ é o valor absoluto da diferença entre a *orientação calculada* θ_{Rcalc} e a *orientação verdadeira* θ_R. Uma vez calculada a orientação do robô, o conhecimento do erro de orientação permitiria determinar duas possíveis soluções para a orientação verdadeira: $\theta_R=\theta_{Rcalc}-\Delta\theta_R$ ou $\theta_R=\theta_{Rcalc}+\Delta\theta_R$ (Figura 5.31).

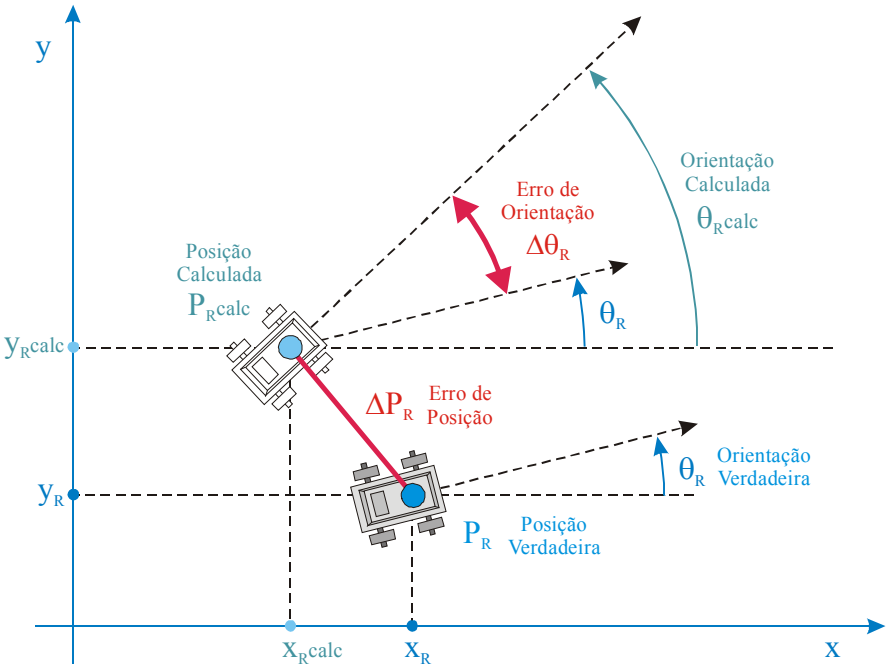


Figura 5.29: Definição *de erro de posição* e de *erro de orientação*.

[17] Considera-se que a posição de um robô de dimensões não desprezáveis é a posição de um dos seus pontos.

Como fontes dos erros de posição e de orientação podem apontar-se:

- Erros grosseiros (por exemplo, os que resultam da incorrecta identificação de uma baliza);

- Erros sistemáticos de medição de ângulos (por exemplo, os que se devem à discrepância entre a posição verdadeira de uma baliza e a posição que lhe é atribuída num mapa conservado na memória do robô);

- Erros aleatórios de medição de ângulos (por exemplo, os que se devem à resolução finita de um instrumento de medição digital);

- Erros devidos a aproximações (por exemplo, os que resultam de aproximar funções não lineares por funções lineares ou à imperfeição dos modelos matemáticos utilizados);

- Erros inerentes aos cálculos.

O valor dos erros de posição e de orientação depende não só do valor dos erros que lhes dão origem mas também da amplificação desses erros feita pelo algoritmo de localização e que, no caso dos métodos que recorrem a balizas, depende da posição do robô relativamente a essas balizas.

Os erros grosseiros são geralmente fáceis de evitar e um algoritmo de localização pode não recorrer a aproximações, mas os erros de medição e os erros de cálculo estão sempre presentes.

Os cálculos podem actualmente ser efectuados – a baixo custo – utilizando o formato em vírgula flutuante IEEE 754 *Double* (64 bits, 53 dos quais significativos), que garante 15 a 17 algarismos decimais significativos (Kahan, 1996; NCG, 1996). Este número de algarismos significativos é muito superior ao que, na triangulação, se requer para a correcta representação das medidas de ângulos[18]. Por isso, é razoável esperar que os efeitos devidos aos erros de cálculo sejam, em geral, desprezáveis face aos provocados pelos erros de medição. De facto, testes realizados recorrendo à simulação

[18] Por exemplo, a correcta representação de medidas de ângulos afectadas de incertezas de ±0,005º requer apenas 5 algarismos decimais.

por computador, parte dos quais se encontra documentada no Capítulo 6, revelam que os erros de posição e de orientação obtidos não são atribuíveis aos erros de cálculo, mesmo quando se trabalha muito perto de singularidades das funções do algoritmo de localização

O valor exacto de um erro de medição é sempre desconhecido, mas geralmente é possível (e necessário) associar a cada medida uma *incerteza de medição*, *"estimativa que caracteriza o intervalo de valores no qual se situa o valor verdadeiro da grandeza medida"* (Almeida, 1997).

Quando é possível estabelecer valores máximos finitos para os erros de medição, as incertezas na posição e na orientação calculadas podem ser caracterizadas por um *erro máximo de posição* e um *erro máximo de orientação*:

- O *erro máximo de posição* $\Delta P_{Rmáx}$ é o valor máximo que o erro de posição pode assumir num dado ponto do plano de navegação quando as medidas dos ângulos variam dentro da gama de valores imposta pela incerteza de medição; corresponde ao raio de um círculo que está centrado na *posição calculada* e contém a *posição verdadeira* (Figura 5.30).

- O *erro máximo de orientação* $\Delta\theta_{Rmáx}$ é o valor máximo que o erro de orientação pode assumir num dado ponto do plano de navegação quando as medidas dos ângulos variam dentro da gama de valores imposta pela incerteza de medição (Figura 5.31).

Neste ponto apresenta-se um método que permite determinar $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. Constitui uma solução para o seguinte problema: *conhecidos os limites dos intervalos dentro dos quais se situam os verdadeiros valores dos ângulos λ_1, λ_{12} e λ_{31}, determinar os valores máximos que o erro de posição e o erro de orientação podem assumir no ponto em que λ_1, λ_{12} e λ_{31} são medidos.* Parte-se do princípio que todos os ângulos medidos pelo robô o são a partir de um mesmo ponto do plano de navegação e que o robô não muda a sua orientação enquanto as medições estão a ser efectuadas. Isto pode ser apenas aproximadamente verdade. No entanto, trata-se de uma aproximação inerente ao método de medição de ângulos e não aos algoritmos que determinam $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$.

Se as fontes de erros sistemáticos forem devidamente identificadas, esses erros podem ser eliminados ou de tal forma reduzidos que, na prática, os seus efeitos são desprezáveis. Mas os erros aleatórios não podem ser totalmente eliminados. Quando não é possível estabelecer um limite máximo finito para estes erros, as incertezas de posição e de orientação que lhes são devidas só podem ser determinadas com níveis de confiança inferiores a 100%.

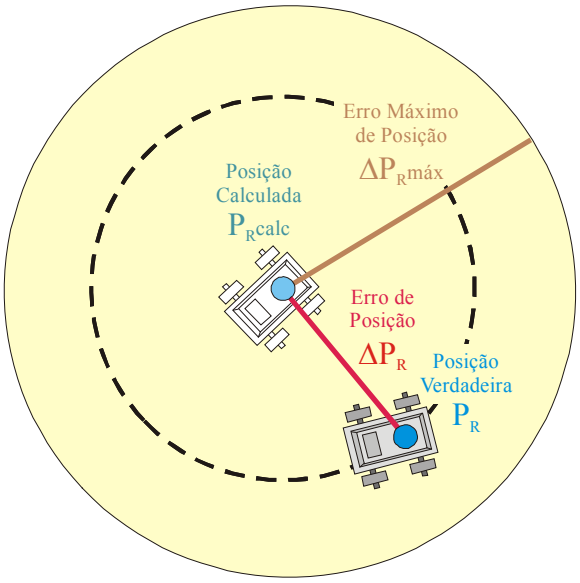


Figura 5.30: O *erro de posição* corresponde ao raio da circunferência que está centrada na *posição calculada* e contém a *posição verdadeira*. O *erro máximo de posição* corresponde ao raio de um círculo que está centrado na *posição calculada* e contém a *posição verdadeira*.

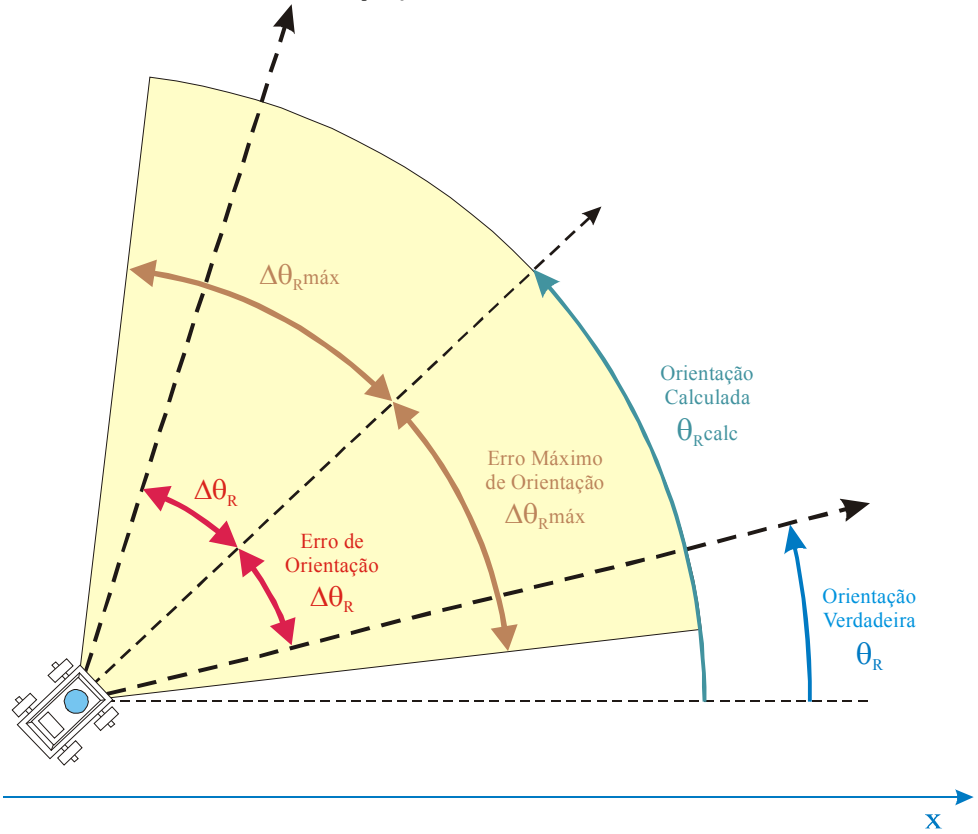


Figura 5.31: O *erro máximo de orientação* é o valor máximo do *erro de orientação.*

Se cada ângulo puder ser medido um número de vezes suficientemente elevado, torna-se possível efectuar o tratamento estatístico das medidas obtidas. Em certos casos, é possível estimar a incerteza de posição devida aos erros aleatórios de medição a partir das incertezas de medição originadas por esses erros. No ponto 5.6.5, indica-se o modo de utilizar com esta finalidade o método desenvolvido para determinar $\Delta P_{Rmáx}$.

Pode não haver verdadeira necessidade ou tempo disponível para efectuar múltiplas medições e o respectivo tratamento estatístico. Os cálculos realizados com base numa única medição de cada ângulo são geralmente mais fáceis de realizar, demoram menos tempo e, muitas vezes, produzem resultados satisfatórios tendo em vista a aplicação em causa. Além disso, se o robô estiver em movimento, à medida que aumenta o número de medições de cada ângulo torna-se cada vez mais difícil de admitir como boa aproximação que todas as medições são efectuadas num mesmo ponto do plano de navegação.

Toda a análise feita até ao ponto 5.6.5 pressupõe que, em cada ponto do plano de navegação, se faz uma única medição de cada ângulo.

5.6.1 Superfície de Incerteza de Medição e Superfície de Incerteza de Posição

Se o ângulo λ_{12} pudesse ser medido por um processo isento de erros de medição, tal processo produziria um valor λ_{12m} que seria coincidente com o verdadeiro valor do ângulo. No plano de navegação seria então possível determinar um arco de circunferência sobre o qual se encontraria a posição verdadeira do robô (Figura 5.32). A medição sem erros de um segundo ângulo λ_{31} produziria uma medida λ_{31m} a partir da qual se poderia determinar um segundo arco de circunferência. Este intersectaria o primeiro na posição verdadeira do robô (Figura 5.33).

Na prática, se λ_{12m} for a medida de λ_{12} resultante de uma medição deste ângulo com uma incerteza $\pm\Delta\lambda$, o valor verdadeiro de λ_{12} está compreendido entre $\lambda_{12m}-\Delta\lambda$ e $\lambda_{12m}+\Delta\lambda$. No plano de navegação é possível determinar uma lúnula[19] (Kuipers e Levitt 1988) cuja fronteira são os arcos de circunferência correspondentes a $\lambda_{12m}-\Delta\lambda$ e $\lambda_{12m}+\Delta\lambda$ e dentro da qual se encontra a posição verdadeira do robô (Figura 5.34).

[19] *Lúnula: domínio plano, com forma de crescente, e cuja fronteira é constituída por dois arcos de circunferência* (DLP, 1994).

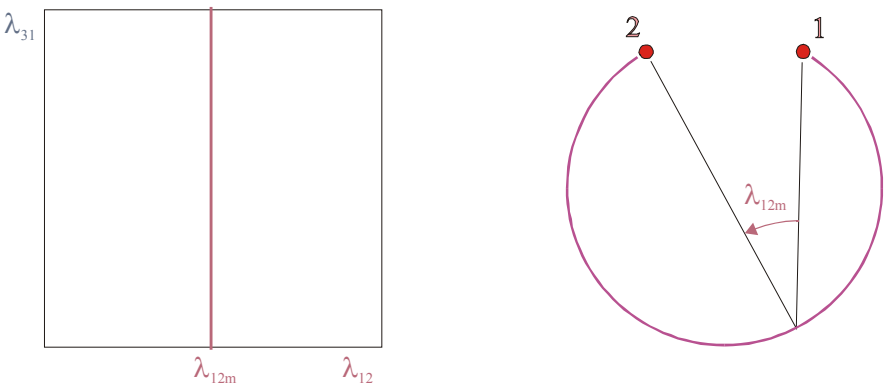


Figura 5.32: Se λ_{12m} coincidisse sempre com o verdadeiro valor de λ_{12}, seria possível determinar no plano de navegação um arco de circunferência sobre o qual se encontraria a posição verdadeira do robô.

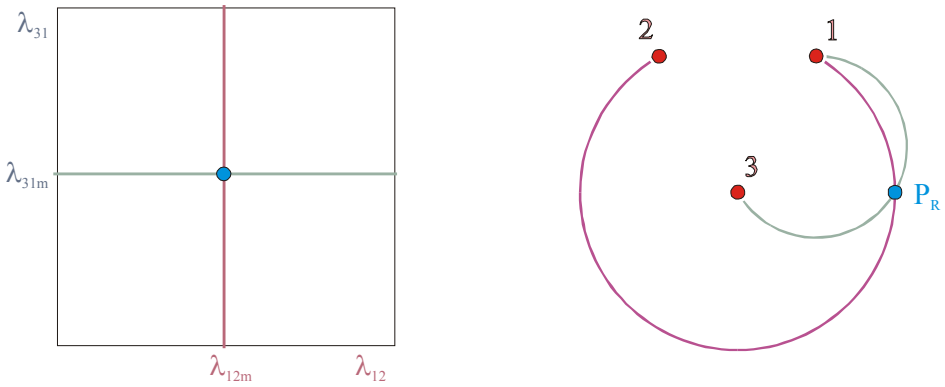


Figura 5.33: Se λ_{12m} e λ_{31m} coincidissem sempre com os verdadeiros valores de λ_{12} e λ_{31}, seria possível determinar a posição verdadeira do robô no plano de navegação, no ponto de intersecção de dois arcos.

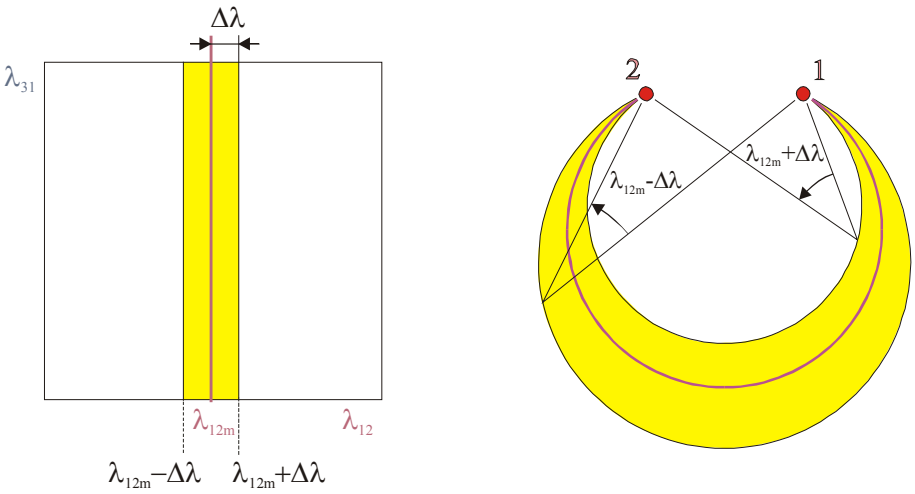


Figura 5.34: Se o valor λ_{12m} for o resultado de uma medição de λ_{12} efectuada com uma incerteza $\pm\Delta\lambda$, a posição verdadeira do robô está dentro de uma lúnula do plano de navegação cujas fronteiras são os arcos de circunferência correspondentes a $\lambda_{12m}-\Delta\lambda$ e $\lambda_{12m}+\Delta\lambda$.

Sendo λ_{12m} e λ_{31m} os resultados de medições independentes de λ_{12} e λ_{31}, é possível restringir a posição do robô à região do plano de navegação em que se intersectam duas lúnulas do plano de navegação (Figura 5.35).

Assume-se que as medições de λ_{12} e λ_{31} possuem ambas a mesma incerteza $\pm\Delta\lambda$, o que é razoável se se partir do princípio que todas as balizas são do mesmo tipo e se está a utilizar um mesmo dispositivo para medir os dois ângulos.

Assim, devido à incerteza de medição de ângulos, deixa de se poder determinar o ponto do plano de navegação que coincide com a posição verdadeira do robô[20]. Mas, se os erros de medição tiverem limites finitos e conhecidos, continua a ser possível definir uma região do plano que contém esse ponto. Doravante chamar-se-á *superfície de incerteza de posição* a essa região e *superfície de incerteza de medição* à sua imagem inversa no referencial $\lambda_{12}0\lambda_{31}$. Estas duas superfícies estão representadas a amarelo na Figura 5.35 e suas congéneres.

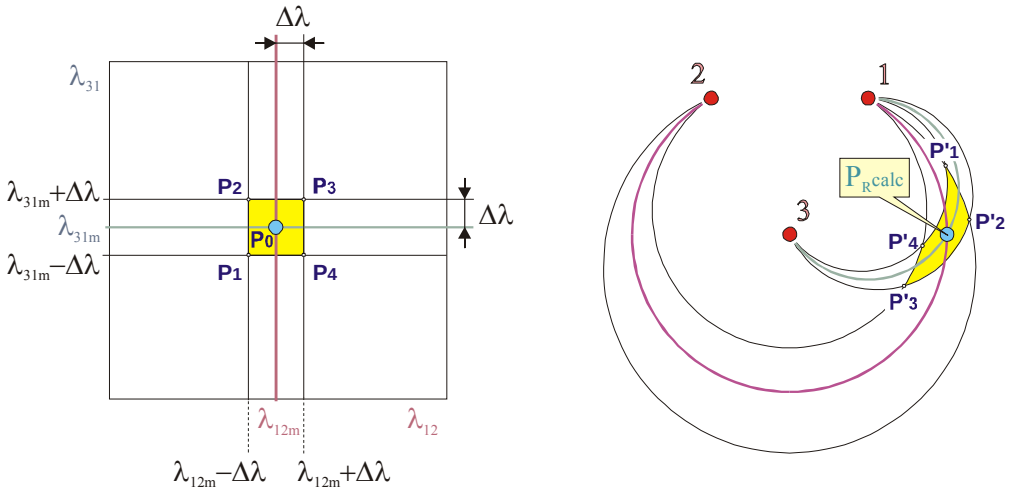


Figura 5.35: Os valores λ_{12m} e λ_{31m} são os resultados das medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$. A posição verdadeira do robô está dentro da superfície correspondente à intersecção de duas lúnulas do plano de navegação.

[20] Recorda-se que a posição verdadeira do robô é a posição de um dos seus pontos.

A observação da Figura 5.35 permite concluir que:

- A posição calculada P_{Rcalc}, situada no interior da *superfície de incerteza de posição,* é a imagem do ponto P_0 cujas coordenadas são λ_{12m} e λ_{31m}, no referencial $\lambda_{12}0\lambda_{31}$;

- A superfície de incerteza de posição possui 4 vértices (P'_1, P'_2, P'_3 e P'_4) que são as imagens dos pontos P_1, P_2, P_3 e P_4 do referencial $\lambda_{12}0\lambda_{31}$. Estes pontos são os vértices da *superfície de incerteza de medição* e possuem as seguintes coordenadas:

$$P_1\ (\lambda_{12m}-\Delta\lambda,\ \lambda_{31m}-\Delta\lambda)$$

$$P_2\ (\lambda_{12m}-\Delta\lambda,\ \lambda_{31m}+\Delta\lambda) \qquad (5.1)$$

$$P_3\ (\lambda_{12m}+\Delta\lambda,\ \lambda_{31m}+\Delta\lambda)$$

$$P_4\ (\lambda_{12m}+\Delta\lambda,\ \lambda_{31m}-\Delta\lambda)$$

Em vez de resultarem de medições dos ângulos λ_{12} e λ_{31}, os valores λ_{12m} e λ_{31m} podem ser calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3 (Figura 5.8). É este o método utilizado quando, por exemplo, as medições de ângulos se realizam com um sistema baseado num sensor rotativo. Os cálculos podem ser efectuados recorrendo ao algoritmo:

1. $\lambda_{12} = \lambda_2 - \lambda_1$
2. Se $\lambda_1 > \lambda_2$ então $\lambda_{12} = \lambda_{12} + 360^o$
3. $\lambda_{31} = \lambda_1 - \lambda_3$
4. Se $\lambda_3 > \lambda_1$ então $\lambda_{31} = \lambda_{31} + 360^o$

Se λ_{1m}, λ_{2m} e λ_{3m} forem as medidas de λ_1, λ_2 e λ_3, e admitindo que a incerteza de medição de ângulos é de $\pm\Delta\lambda$, os valores verdadeiros de λ_1, λ_2 e λ_3 estão contidos, respectivamente, nos intervalos $\lambda_{1m}\pm\Delta\lambda$, $\lambda_{2m}\pm\Delta\lambda$ e $\lambda_{3m}\pm\Delta\lambda$. Então, como resultado das

subtracções efectuadas nos passos 1 e 3 do algoritmo, os valores verdadeiros de λ_{12} e λ_{31} estão contidos, respectivamente, nos intervalos $\lambda_{12m}\pm2\Delta\lambda$ e $\lambda_{31m}\pm2\Delta\lambda$.

No entanto, a situação é diferente da que resultaria de medições independentes dos ângulos λ_{12} e λ_{31} com uma incerteza de medição $\pm2\Delta\lambda$ (Figura 5.36). De facto, ao calcular λ_{12m} e λ_{31m} a partir de medições dos ângulos λ_1, λ_2 e λ_3 estabelece-se uma dependência entre os dois resultados, uma vez que a medida de λ_3 intervém – e com sinais contrários – nos cálculos de ambos os valores. Devido a esta dependência, a incerteza associada a $\lambda_{12m} + \lambda_{31m}$ é também $\pm2\Delta\lambda$ (e não $\pm4\Delta\lambda$, como ocorreria se tivessem sido efectuadas medições independentes de λ_{12} e λ_{31} com uma incerteza de medição $\pm2\Delta\lambda$). Assim, a superfície de incerteza de posição é, neste caso, a intersecção das duas lúnulas representadas na Figura 5.36 com uma terceira lúnula[21] cuja fronteira são os arcos de circunferência[22] correspondentes a $\lambda_{12m} + \lambda_{31m}-2\Delta\lambda$ e $\lambda_{12m} + \lambda_{31m}+2\Delta\lambda$ (Figura 5.37).

[21] Kuipers e Levitt (1988) e Garulli e Vicino (2001) referem que a superfície de incerteza de posição é a intersecção de várias lúnulas. Briechle e Hanebeck (2004) recorrem à intersecção de três lúnulas, resultantes da medição de três ângulos. Na análise feita por Sutherland (1994) também se constata que a superfície de incerteza de posição é a intersecção de duas ou três lúnulas. No entanto, considera-se que o número de lúnulas que se intersectam depende da posição do robô relativamente às balizas (ver também Sutherland e Thompson, 1994). Segundo a autora, a superfície de incerteza de posição:

- é a intersecção de duas lúnulas se as três balizas forem colineares ou enquanto o robô permanecer fora do triângulo formado por três balizas não colineares;
- transforma-se na intersecção de três lúnulas quando o robô se encontra dentro desse triângulo.

Não estamos de acordo com estas conclusões ou, pelo menos, com a sua generalidade. Boa parte das situações representadas nas figuras que ilustram este capítulo constituem contra-exemplos dessas deduções. Por exemplo, a superfície de incerteza representada na Figura 5.37 é a intersecção de três lúnulas apesar de o robô estar fora do triângulo formado pelas balizas. E, na Figura 5.38, todas as superfícies de incerteza representadas são intersecções de duas lúnulas, independentemente de se encontrarem dentro ou fora do triângulo formado pelas balizas. De acordo com o exposto neste capítulo, o número de lúnulas cuja intersecção é a superfície de incerteza depende do modo como os ângulos λ_{12m} e λ_{31m} são obtidos e não da posição do robô relativamente às balizas.

[22] Às rectas $\lambda_{12}+\lambda_{31} = k$ ($0º<k<720º$) no referencial $\lambda_{12}0\lambda_{31}$ correspondem, no plano de navegação, arcos de circunferência cujas extremidades se situam nas balizas 2 e 3. De facto, segundo a análise feita no ponto 5.2, verifica-se sempre que $\lambda_{12}+\lambda_{31} = 360º-\lambda_{23}$ ou então $\lambda_{12}+\lambda_{31} = 720º-\lambda_{23}$. Uma vez que a λ_{23} correspondem, no plano de navegação, arcos de circunferência cujas extremidades se situam nas balizas 2 e 3, então o mesmo sucede a $\lambda_{12}+\lambda_{31}$.

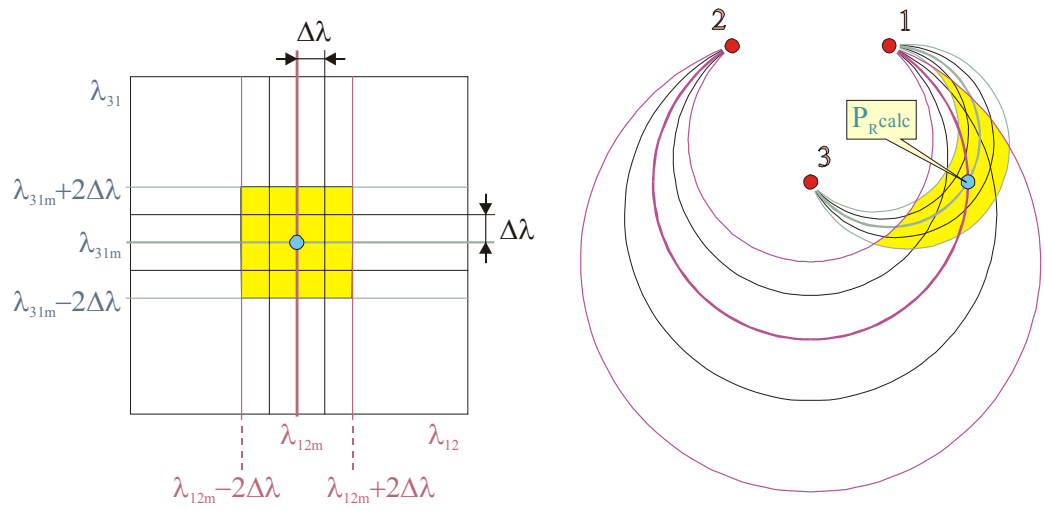


Figura 5.36: Os valores λ_{12m} e λ_{31m} são os resultados das medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm 2\Delta\lambda$. A superfície de incerteza de posição é a intersecção de duas lúnulas do plano de navegação.

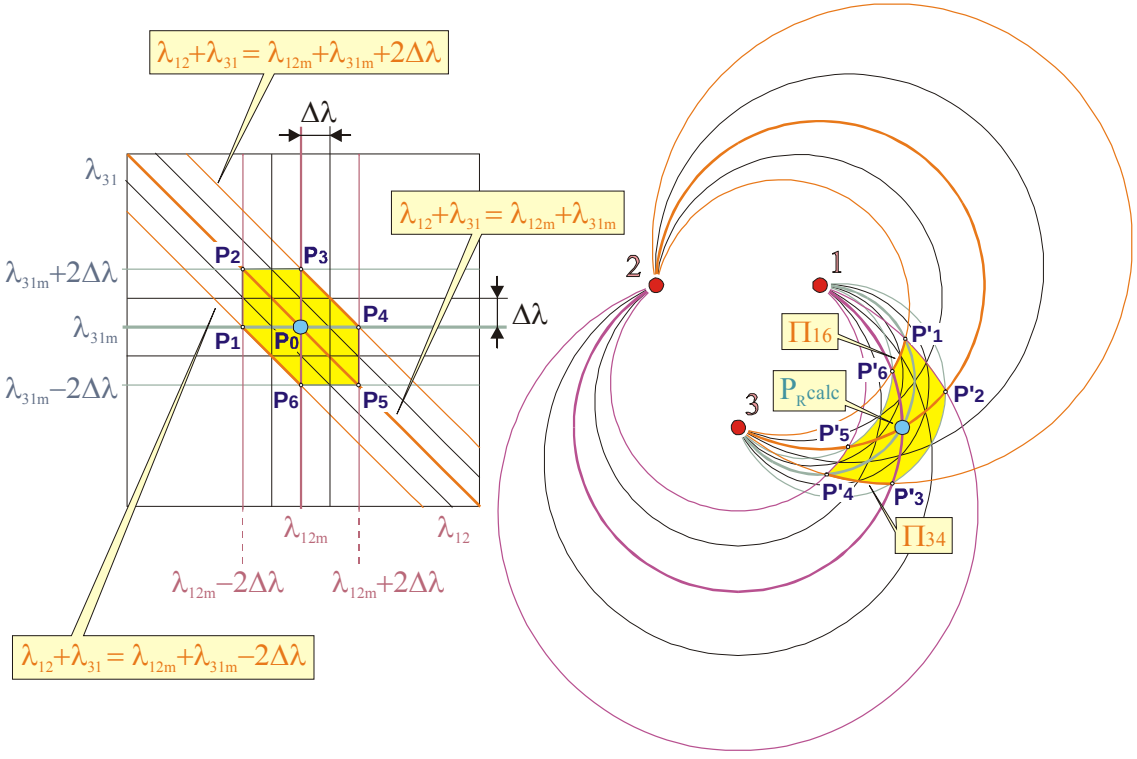


Figura 5.37: Os valores λ_{12m} e λ_{31m} são calculados a partir de medições independentes de λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$. A superfície de incerteza de posição é a intersecção de três lúnulas do plano de navegação. Os arcos Π_{16} e Π_{34} unem o ponto P'_1 ao ponto P'_6 e o ponto P'_3 ao ponto P'_4, respectivamente. São as imagens dos segmentos de recta que, no referencial $\lambda_{12}0\lambda_{31}$, unem o ponto P_1 ao ponto P_6 e o ponto P_3 ao ponto P_4, respectivamente.

A observação da Figura 5.37 permite concluir que:

- A posição calculada P_{Rcalc}, situada no interior da *superfície de incerteza de posição,* é a imagem do ponto P_0 cujas coordenadas são λ_{12m} e λ_{31m}, no referencial $\lambda_{12}0\lambda_{31}$;

- A superfície de incerteza de posição possui 6 vértices (P'_1, P'_2, P'_3, P'_4, P'_5 e P'_6) que são as imagens dos pontos P_1, P_2, P_3, P_4, P_5 e P_6 do referencial $\lambda_{12}0\lambda_{31}$. Estes pontos são os vértices da *superfície de incerteza de medição* e possuem as seguintes coordenadas:

$$\begin{aligned} &P_1\ (\lambda_{12m}-2\Delta\lambda,\ \lambda_{31m}) \\ &P_2\ (\lambda_{12m}-2\Delta\lambda,\ \lambda_{31m}+2\Delta\lambda) \\ &P_3\ (\lambda_{12m},\ \lambda_{31m}+2\Delta\lambda) \\ &P_4\ (\lambda_{12m}+\Delta\lambda,\ \lambda_{31m}) \\ &P_5\ (\lambda_{12m}+\Delta\lambda,\ \lambda_{31m}-\Delta\lambda) \\ &P_6\ (\lambda_{12m},\ \lambda_{31m}-\Delta\lambda) \end{aligned} \qquad (5.2)$$

Resumindo: a *superfície de incerteza de medição* é um polígono situado no referencial $\lambda_{12}0\lambda_{31}$ que, dependendo do modo como os ângulos λ_{12m} e λ_{31m} são obtidos, pode ter 4 ou 6 vértices. Os seus lados são segmentos de recta. A sua imagem no plano de navegação é a *superfície de incerteza de posição*, que contém a *posição calculada* e também – se os erros de medição tiverem limites finitos e conhecidos – a *posição verdadeira* do robô. É um polígono com o mesmo número de vértices que a respectiva superfície de incerteza de medição e cujos lados são arcos de circunferência.

Como se pode ver na Figura 5.38, a mesma incerteza nas medições independentes de λ_{12} e λ_{31} realizadas em diversos pontos do plano de navegação produz superfícies de incerteza de posição cujas dimensões dependem do ponto em que as medições são realizadas. A variação das dimensões da superfície de incerteza de posição com o ponto em que as medições são realizadas, para uma dada incerteza de medição, é um fenómeno comum a todos os sistemas de localização com balizas (Drane e Rizos, 1998).

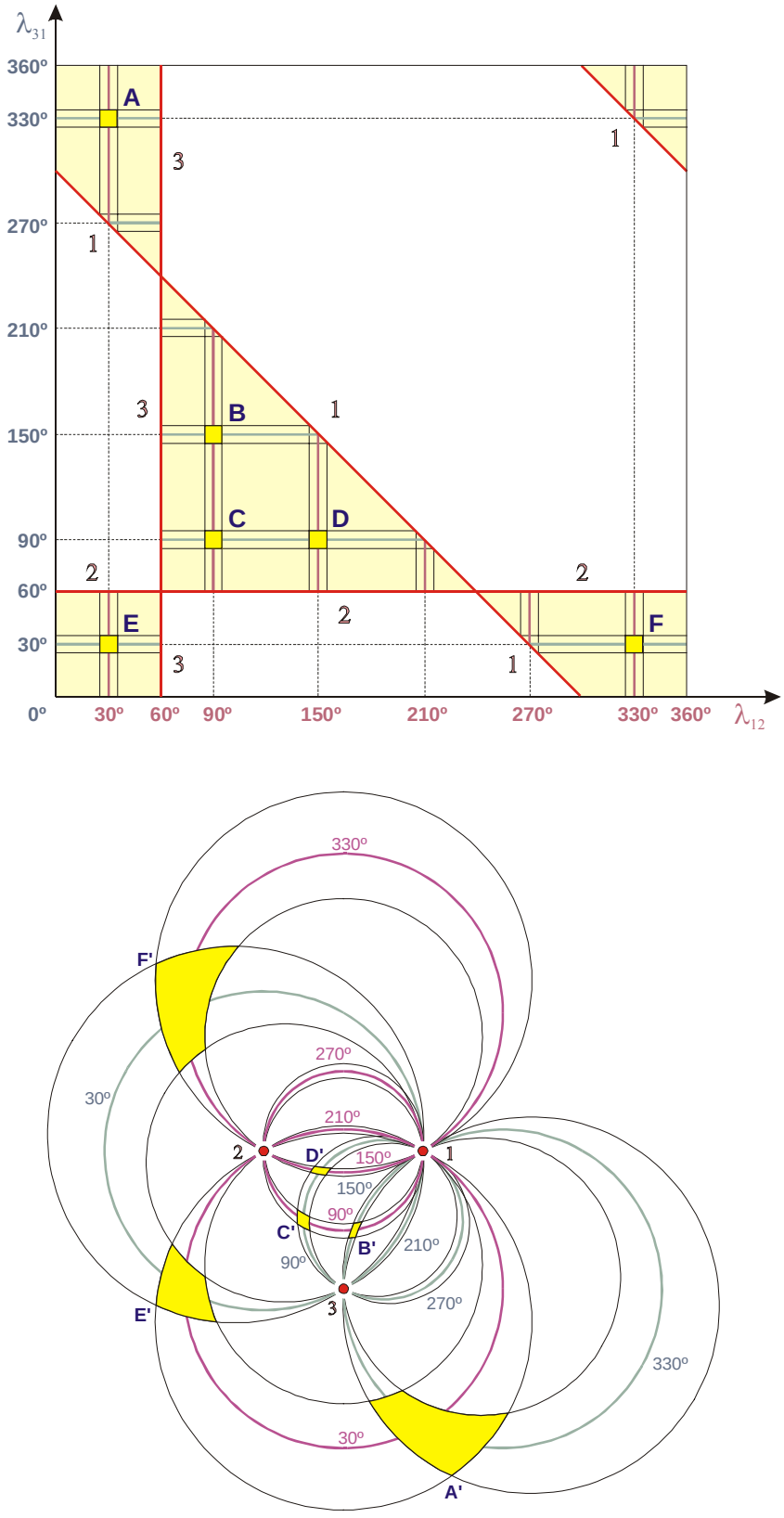


Figura 5.38: A mesma incerteza (±5º) nas medições independentes de λ_{12} e λ_{31} realizadas em diversos pontos do plano de navegação produz superfícies de incerteza de posição cujas dimensões dependem do ponto em que as medições são realizadas.

5.6.2 Determinação do Erro Máximo de Posição

Sejam $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ duas funções de λ_{12} e λ_{31} utilizadas para calcular a posição do robô no referencial x0y definido no plano de navegação. As coordenadas x e y de cada ponto do plano são dadas por

$$\begin{aligned} x &= f(\lambda_{12}, \lambda_{31}) \\ y &= g(\lambda_{12}, \lambda_{31}) \end{aligned} \tag{5.3}$$

Sejam λ_{12m} e λ_{31m} as medidas de λ_{12} e λ_{31}. As coordenadas x_{Rcalc} e y_{Rcalc} da posição calculada do robô P_{Rcalc} são dadas por

$$\begin{aligned} x_{Rcalc} &= f(\lambda_{12m}, \lambda_{31m}) \\ y_{Rcalc} &= g(\lambda_{12m}, \lambda_{31m}) \end{aligned} \tag{5.4}$$

Sejam Δx_R e Δy_R tais que

$$\begin{aligned} \Delta x_R &= x_R - x_{Rcalc} \\ \Delta y_R &= y_R - y_{Rcalc} \end{aligned} \tag{5.5}$$

em que x_R e y_R são as coordenadas da posição verdadeira do robô, P_R.

Sejam $\pm\Delta\lambda_{12}$ e $\pm\Delta\lambda_{31}$ as incertezas de medição associadas a λ_{12m} e λ_{31m}. Se for válido aproximar as funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ pelas respectivas séries de Taylor de primeira ordem na vizinhança do ponto $(\lambda_{12m}, \lambda_{31m})$, para calcular $\Delta x_{Rmáx}$ e $\Delta y_{Rmáx}$ (valores máximos de Δx_R e Δy_R) pode recorrer-se às expressões (Mcgillem e Rappaport, 1989; Taylor, 1997):

$$\begin{aligned} \Delta x_R \le \Delta x_{Rmáx} &\approx \Delta\lambda_{12} \cdot \left| \frac{\partial f(\lambda_{12}, \lambda_{31})}{\partial \lambda_{12}} \right|_{\substack{\lambda_{12}=\lambda_{12m} \\ \lambda_{31}=\lambda_{31m}}} + \Delta\lambda_{31} \cdot \left| \frac{\partial f(\lambda_{12}, \lambda_{31})}{\partial \lambda_{31}} \right|_{\substack{\lambda_{12}=\lambda_{12m} \\ \lambda_{31}=\lambda_{31m}}} \\ \Delta y_R \le \Delta y_{Rmáx} &\approx \Delta\lambda_{12} \cdot \left| \frac{\partial g(\lambda_{12}, \lambda_{31})}{\partial \lambda_{12}} \right|_{\substack{\lambda_{12}=\lambda_{12m} \\ \lambda_{31}=\lambda_{31m}}} + \Delta\lambda_{31} \cdot \left| \frac{\partial g(\lambda_{12}, \lambda_{31})}{\partial \lambda_{31}} \right|_{\substack{\lambda_{12}=\lambda_{12m} \\ \lambda_{31}=\lambda_{31m}}} \end{aligned} \tag{5.6}$$

O erro máximo de posição $\Delta P_{Rmáx}$ é então dado por

$$\Delta P_{Rmáx} = \sqrt{\Delta x_{Rmáx}^2 + \Delta y_{Rmáx}^2} \tag{5.7}$$

O método descrito possui as seguintes limitações:

- Os valores de $\Delta x_{Rmáx}$ e $\Delta y_{Rmáx}$ são aproximados, uma vez que as funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ são aproximadas pelas respectivas séries de Taylor de primeira ordem na vizinhança do ponto $(\lambda_{12m}, \lambda_{31m})$;

- As expressões analíticas das derivadas parciais das funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ em ordem a λ_{12} e a λ_{31} podem ser extremamente difíceis de obter, tornando-se necessário efectuar mais aproximações num método que já é aproximado por natureza;

- Devido às aproximações inerentes ao método, este só é válido se $\Delta\lambda_{12}$ e $\Delta\lambda_{31}$ forem suficientemente pequenos.

Um segundo método de estimar a incerteza de posição (Kelly, 1996) recorre ao determinante de [**J**], matriz Jacobiana das funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$:

$$[\mathbf{J}] = \begin{bmatrix} \dfrac{\partial f(\lambda_{12},\lambda_{31})}{\partial \lambda_{12}} & \dfrac{\partial f(\lambda_{12},\lambda_{31})}{\partial \lambda_{31}} \\ \dfrac{\partial g(\lambda_{12},\lambda_{31})}{\partial \lambda_{12}} & \dfrac{\partial g(\lambda_{12},\lambda_{31})}{\partial \lambda_{31}} \end{bmatrix} \tag{5.8}$$

O determinante de [**J**] chama-se *Jacobiano* das funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ e representa-se por:

$$J = \begin{vmatrix} \dfrac{\partial f(\lambda_{12},\lambda_{31})}{\partial \lambda_{12}} & \dfrac{\partial f(\lambda_{12},\lambda_{31})}{\partial \lambda_{31}} \\ \dfrac{\partial g(\lambda_{12},\lambda_{31})}{\partial \lambda_{12}} & \dfrac{\partial g(\lambda_{12},\lambda_{31})}{\partial \lambda_{31}} \end{vmatrix} \tag{5.9}$$

Seja S a área da superfície de incerteza de medição (referencial $\lambda_{12}0\lambda_{31}$), S' a área da respectiva superfície de incerteza de posição (referencial x0y) e J_m o valor de J no ponto $(\lambda_{12m}, \lambda_{31m})$ do referencial $\lambda_{12}0\lambda_{31}$. Pode provar-se (Piskounov, 1997) que

$$S' \approx |J_m| \cdot S \tag{5.10}$$

Se os valores λ_{12m} e λ_{31m} forem os resultados das medições independentes de λ_{12} e λ_{31}, ambas efectuadas com uma incerteza $\pm\Delta\lambda$ (ou seja, $\Delta\lambda_{12} = \Delta\lambda_{31} = \Delta\lambda$), então

$$S = (2\Delta\lambda)^2 \tag{5.11}$$

e verifica-se que

$$S' \approx |J_m| \cdot (2\Delta\lambda)^2 \tag{5.12}$$

Quanto maior for o módulo de J_m, maior é a área da superfície de incerteza de posição, para uma dada incerteza de medição associada a λ_{12m} e λ_{31m}. Esta propriedade pode ser utilizada para estimar o erro de posição.

Este segundo método também possui limitações importantes:

- O valor de S' é apenas aproximado, uma vez que a superfície de incerteza de posição é aproximada por um paralelogramo (Piskounov, 1997).

- O Jacobiano pode ter de ser calculado a partir das derivadas parciais das funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ em ordem a λ_{12} e a λ_{31}. Como já se referiu, as expressões analíticas destas derivadas podem ser extremamente difíceis de obter. Mais uma vez se coloca o problema de ter de introduzir novas aproximações num método que já é aproximado por natureza.

- Devido às aproximações inerentes ao método, este só é válido se $\Delta\lambda_{12}$ e $\Delta\lambda_{31}$ forem suficientemente pequenos.

- Não é garantido - pelo menos em princípio - que o erro de posição seja pequeno só pelo facto de a área da superfície de incerteza de posição ser pequena.

- O método pressupõe que as variáveis de entrada são independentes, o que não é verdade se λ_{12m} e λ_{31m} forem calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3.

O método que seguidamente se sugere para calcular o valor máximo que o erro de posição pode assumir em cada posição do plano de navegação, para uma dada incerteza de medição de ângulos, não possui nenhuma das limitações que caracterizam os dois métodos anteriores.

O *erro máximo de posição*, valor máximo do *erro de posição,* é a distância entre a posição calculada e o ponto da superfície de incerteza de posição que se encontra mais afastado da posição calculada (Figura 5.39 e Figura 5.40). Obviamente, este ponto situa-se no contorno da superfície, havendo duas possibilidades:

1. O ponto da superfície de incerteza de posição que se encontra mais afastado da posição calculada é um vértice dessa superfície (Figura 5.39 e Figura 5.40);

2. O ponto da superfície de incerteza de posição que se encontra mais afastado da posição calculada não é um vértice dessa superfície, mas pertence a um arco de circunferência cujas extremidades se situam em vértices da superfície.

Seguidamente analisar-se-ão as duas possibilidades apontadas e as circunstâncias em que cada uma delas acontece.

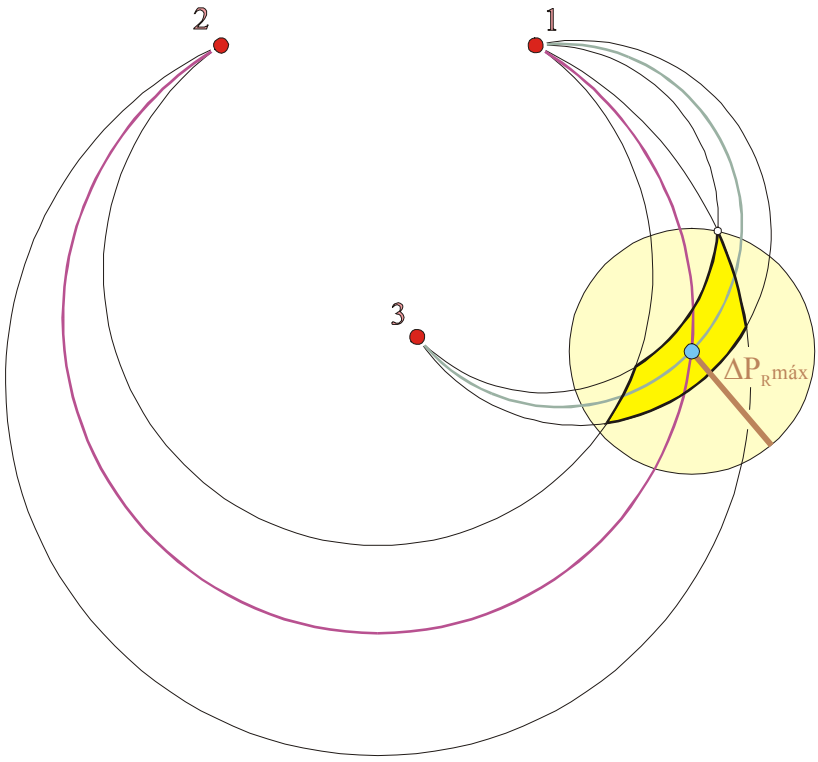


Figura 5.39: Erro máximo de posição numa superfície de incerteza de posição que resulta de se realizarem medições independentes de λ_{12} e λ_{31}.

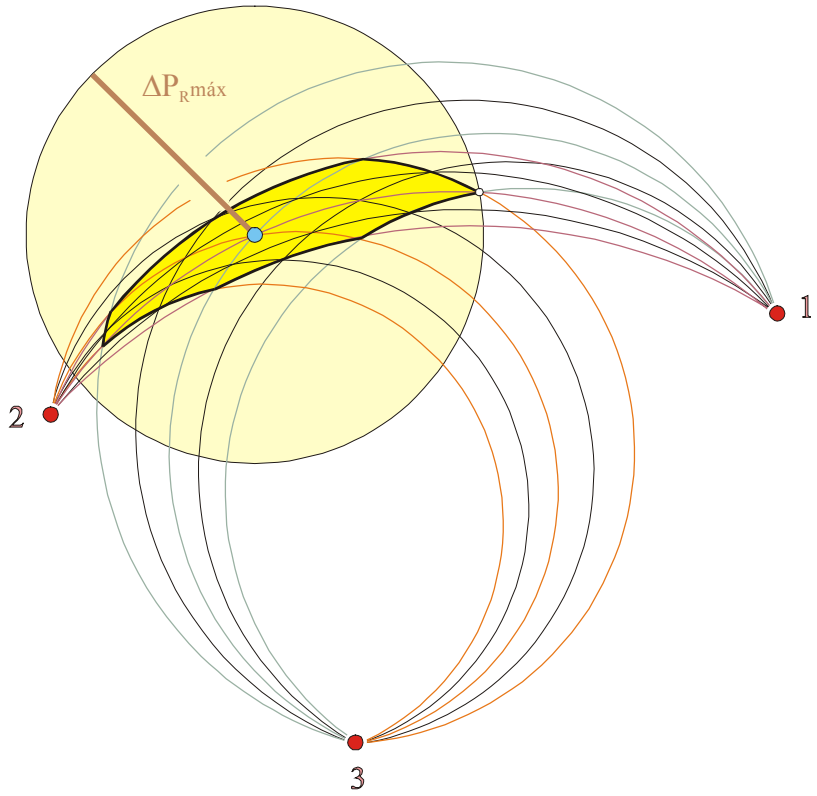


Figura 5.40: Erro máximo de posição numa superfície de incerteza de posição que resulta de λ_{12m} e λ_{31m} serem calculados a partir de medições independentes de λ_1, λ_2 e λ_3.

Na Figura 5.41, P_L e P_P são dois vértices da superfície de incerteza de posição e P_{Rcalc} é a posição calculada. Π é o arco de circunferência que une os dois pontos. Faz parte do contorno da superfície de incerteza de posição e é uma parte do arco de circunferência cujas extremidades são as balizas A e B. P_L está mais longe de P_{Rcalc} do que P_P. Λ é a circunferência centrada em P_{Rcalc} que contém P_L e $d_{máx}$ é a distância máxima de P_{Rcalc} a Π. Surgem, então, duas possibilidades:

- Se Π não intersectar Λ (Figura 5.41 e Figura 5.42), $d_{máx}$ é igual à distância entre P_{Rcalc} e P_L (ou seja, P_L é o ponto de Π que fica mais afastado de P_{Rcalc}).

- Se Π intersectar Λ (Figura 5.43), $d_{máx}$ é superior à distância entre P_{Rcalc} e P_L (ou seja, P_L não é o ponto de Π que fica mais afastado de P_{Rcalc}). Neste caso, $d_{máx}$ é sempre igual à distância entre P_{Rcalc} e C mais o valor de R (C e R são, respectivamente, o centro e o raio da circunferência que contém Π)[23].

[23] A distância máxima de uma circunferência a um ponto que lhe é interior é o comprimento de um segmento de recta que tem uma extremidade no ponto, a outra extremidade sobre a circunferência e contém o seu centro.

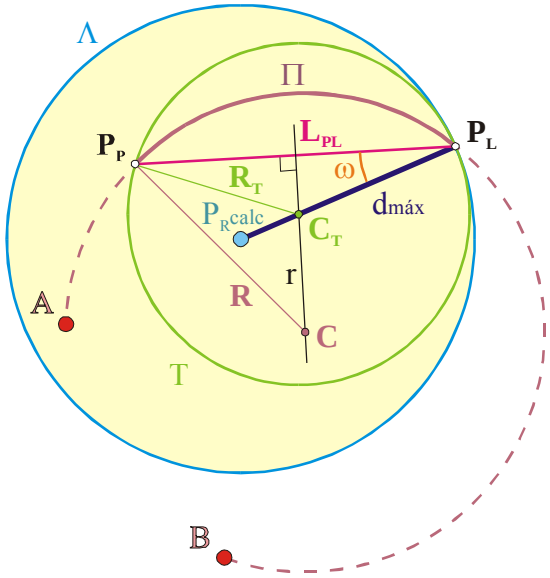


Figura 5.41: Se as balizas A e B se situarem no exterior da circunferência T, P_L é o ponto de Π que fica mais afastado de P_{Rcalc}.

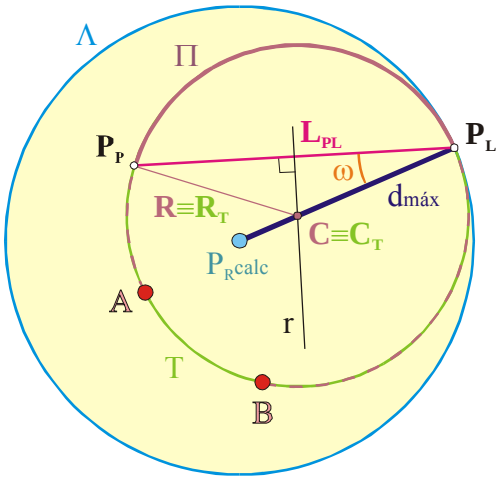


Figura 5.42: Se as balizas A e B se situarem sobre a circunferência T, P_L é o ponto de Π que fica mais afastado de P_{Rcalc}.

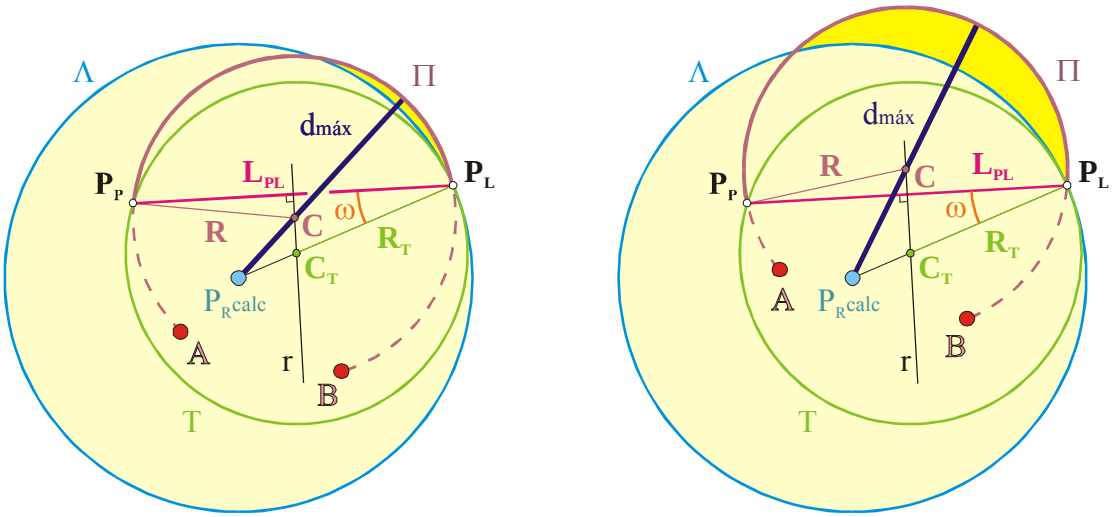


Figura 5.43: Se as balizas A e B se situarem no exterior da circunferência T, P_L não é o ponto de Π que fica mais afastado de P_{Rcalc}.

Resta verificar em que condições é que Π intersecta Λ. Para começar, considere-se a circunferência T com centro C_T e raio R_T (Figura 5.41, Figura 5.42 e Figura 5.43). T contém P_L e P_P e é tangente, no ponto P_L, à circunferência Λ. Seguidamente, faça-se o seguinte raciocínio:

a) Nenhum dos pontos P_L e P_P coincide com uma baliza;

b) O arco Π é contínuo entre P_L e P_P porque é uma parte do contorno da superfície de incerteza de posição, que é uma superfície fechada;

c) Das considerações anteriores deduz-se que as balizas A e B não podem pertencer ao arco Π, por isso não pode acontecer que uma das balizas seja interior à circunferência T e a outra lhe seja exterior: as balizas têm de ser ambas exteriores a T (Figura 5.41), ambas pertencentes a T (Figura 5.42) ou então ambas interiores a T (Figura 5.43);

d) Se as balizas A e B estiverem no interior da circunferência T, então o arco Π está no exterior de T e intersecta a circunferência Λ;

e) Se as balizas A e B estiverem sobre ou no exterior da circunferência T, então o arco Π está sobre ou no interior de T que, por definição, é sempre interior à circunferência Λ.

De acordo com o que foi exposto, é possível concluir: para que P_L seja o ponto do arco Π que está mais afastado de P_{Rcalc} , a distância entre C_T e qualquer uma das balizas A ou B tem de ser igual ou superior a R_T (assim, as balizas estão sobre ou no exterior da circunferência T). Esta condição é necessária e suficiente [24].

Sugere-se o seguinte algoritmo para determinar o erro máximo de posição $\Delta P_{Rmáx}$:

[24] Também é possível concluir que P_L é o ponto do arco Π que está mais afastado de P_{Rcalc} se se verificar alguma das condições, que são suficientes mas não necessárias:

- a distância entre as balizas A e B é igual ou superior a $2R_T$ (as balizas não podem estar no interior da circunferência T);
- a distância entre P_{Rcalc} e qualquer uma das balizas A ou B é superior à distância entre P_{Rcalc} e P_L (alguma das balizas está no exterior da circunferência Λ e, portanto, no exterior da circunferência T).

1. Determinar as coordenadas de P_{Rcalc} no referencial x0y definido no plano de navegação, utilizando as coordenadas de P_0 no referencial $\lambda_{12}0\lambda_{31}$ (λ_{12m} e λ_{31m}) como entradas do algoritmo de triangulação;

2. Determinar as coordenadas de cada um dos vértices da superfície de incerteza de posição (referencial x0y). Para cada vértice utilizam-se como entradas do algoritmo de triangulação as coordenadas da imagem inversa desse vértice, no referencial $\lambda_{12}0\lambda_{31}$;

3. Determinar as distâncias entre a posição calculada e cada vértice da superfície de incerteza de posição;

4. Fazer $\Delta P_{Rmáx}$ igual à maior das distâncias calculadas no ponto anterior;

5. Determinar as distâncias entre cada par de vértices da superfície de incerteza de posição pertencentes ao mesmo arco;

6. Para cada par de vértices da superfície de incerteza de posição pertencentes ao mesmo arco (cujas extremidades se situam nas balizas A e B):

    a. determinar qual dos dois vértices está mais longe de P_{Rcalc} (seja P_L esse vértice e P_P o outro);

    b. determinar ω, ângulo convexo formado pelo segmento de recta que une os pontos P_P e P_L com o segmento de recta que une os pontos P_{Rcalc} e P_L (ω pode ser determinado a partir das coordenadas de P_{Rcalc}, P_L e P_P no referencial x0y);

    c. determinar R_T, raio da circunferência T, utilizando a expressão: $R_T = \frac{\frac{L_{PL}}{2}}{\cos\omega}$, em que L_{PL} é a distância existente entre P_P e P_L;

    d. determinar as coordenadas de C_T, centro da circunferência T, no referencial x0y, a partir de R_T e das coordenadas de P_{Rcalc} e P_L no referencial x0y[25];

    e. se a distância entre C_T e alguma das balizas A ou B for inferior a R_T

        fazer $d_{máx}$ igual à distância entre P_{Rcalc} e C_T mais R_T;

        se $\Delta P_{Rmáx} < d_{máx}$ então $\Delta P_{Rmáx} = d_{máx}$;

Este algoritmo possui as seguintes características:

a) Funciona com todos os algoritmos de autolocalização absoluta por triangulação com três balizas (é independente do algoritmo de triangulação utilizado).

b) É exacto (não há aproximações que lhe sejam inerentes).

[25] P_{Rcalc}, C_T e P_L estão sobre a mesma recta e R_T é a distância entre C_T e P_L.

c) Por ser exacto, não requer que a incerteza de medição de ângulos se mantenha abaixo de um determinado limiar.

d) Não requer o cálculo de derivadas parciais difíceis de obter.

e) Pode utilizar-se quando λ_{12m} e λ_{31m} resultam de medições independentes dos ângulos λ_{12} e λ_{31} ou quando são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3.

f) Só deve ser utilizado se o algoritmo de triangulação usado no cálculo da posição for suficientemente rápido uma vez que, para cada posição calculada do robô, esse algoritmo tem de ser executado

- 5 vezes se λ_{12m} e λ_{31m} resultarem de medições independentes dos ângulos λ_{12} e λ_{31};
- 7 vezes se λ_{12m} e λ_{31m} forem calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3.

5.6.3 Determinação do Erro Máximo de Orientação

Como se viu anteriormente, para calcular a orientação do robô é possível utilizar-se o seguinte algoritmo:

1. $\theta_R = \phi + \tau - \lambda_1$
2. Se $\theta_R \leq -180^o$ então $\theta_R = \theta_R + 360^o$
3. Se $\theta_R > 180^o$ então $\theta_R = \theta_R - 360^o$

As operações efectuadas nas linhas 2 e 3 não aumentam a incerteza associada a θ_R porque são uma soma e uma subtracção de uma quantidade exacta[26] ao valor previamente calculado. Uma vez que ϕ é calculado a partir das posições conhecidas *a priori* das balizas 1 e 2 e não a partir de medições, o valor obtido não contém incerteza devida a medições. Assim, a incerteza $\pm\Delta\theta_{max}$ associada ao valor calculado de θ_R é devida apenas às incertezas $\pm\Delta\tau_{max}$ e $\pm\Delta\lambda$ associadas, respectivamente, a τ_{calc} (valor

[26] Se as operações forem efectuadas em radianos, utiliza-se 2π em vez de 360º, e π é uma constante não exacta. No entanto, π pode ser representado com um número de algarismos muito superior aos algarismos significativos necessários para representar convenientemente os ângulos τ e λ_1. Pode assim desprezar-se o aumento da incerteza associada a θ_R originado pela representação de π com um número finito de algarismos.

calculado de τ) e a λ_{1m}. Se os verdadeiros valores de τ e λ_1 se situarem dentro dos intervalos $\tau_{calc}\pm\Delta\tau_{max}$ e $\lambda_{1m}\pm\Delta\lambda$, então $\Delta\theta_{max}$ não pode ser superior a $\Delta\tau_{max}+\Delta\lambda$. Como $\Delta\lambda$ é um dado do problema do cálculo do erro máximo de orientação, resta determinar $\Delta\tau_{max}$.

O ângulo τ_{calc} é orientado e tal que $-180^o < \tau_{calc} \leq 180^o$ (Figura 5.44, Figura 5.45 e Figura 5.46). O seu lado origem é o segmento de recta que une as balizas 1 e 2. O lado extremidade é o segmento de recta que une a posição calculada P_{Rcalc} e a baliza 1. Em geral, este ângulo não coincide com τ, previamente definido, porque P_{Rcalc} geralmente também não coincide com a posição verdadeira do robô. $\Delta\tau$ é o valor absoluto da diferença $\tau_{calc}-\tau$, e $\Delta\tau_{max}$ é o valor máximo que $\Delta\tau$ pode assumir dentro de uma dada superfície de incerteza de posição. É o maior ângulo convexo[27] que o segmento que une a baliza 1 a P_{Rcalc} pode formar com um segmento que una a baliza 1 a um ponto da superfície de incerteza de posição. Tal ponto $P_{\Delta\tau máx}$ da superfície tem de se situar no seu contorno. Pode haver dois pontos ($P_{\Delta\tau máx}$ e $P'_{\Delta\tau máx}$) do contorno da superfície de incerteza de posição nos quais ocorrem $\Delta\tau_{max}$, como se pode observar na Figura 5.45.

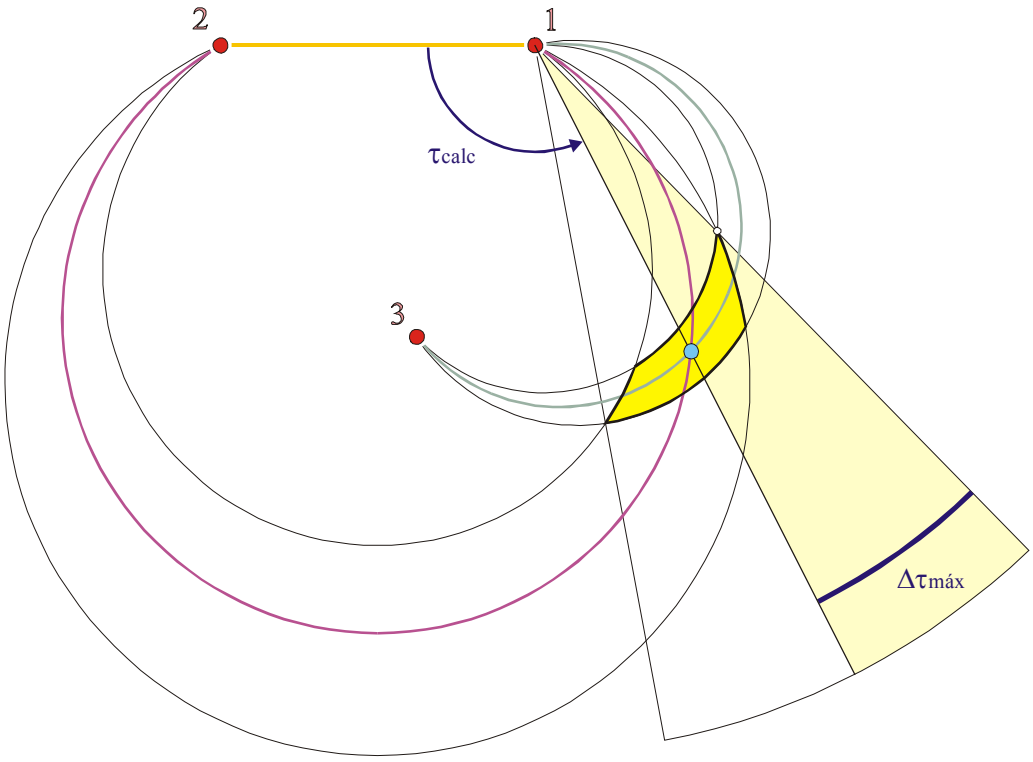


Figura 5.44: $\Delta\tau_{máx}$ numa superfície de incerteza de posição que resulta de se realizarem medições independentes de λ_{12} e λ_{31}.

[27] Não é concebível que, na prática, $\Delta\tau_{max}$ possa assumir valores iguais ou superiores a 180º.

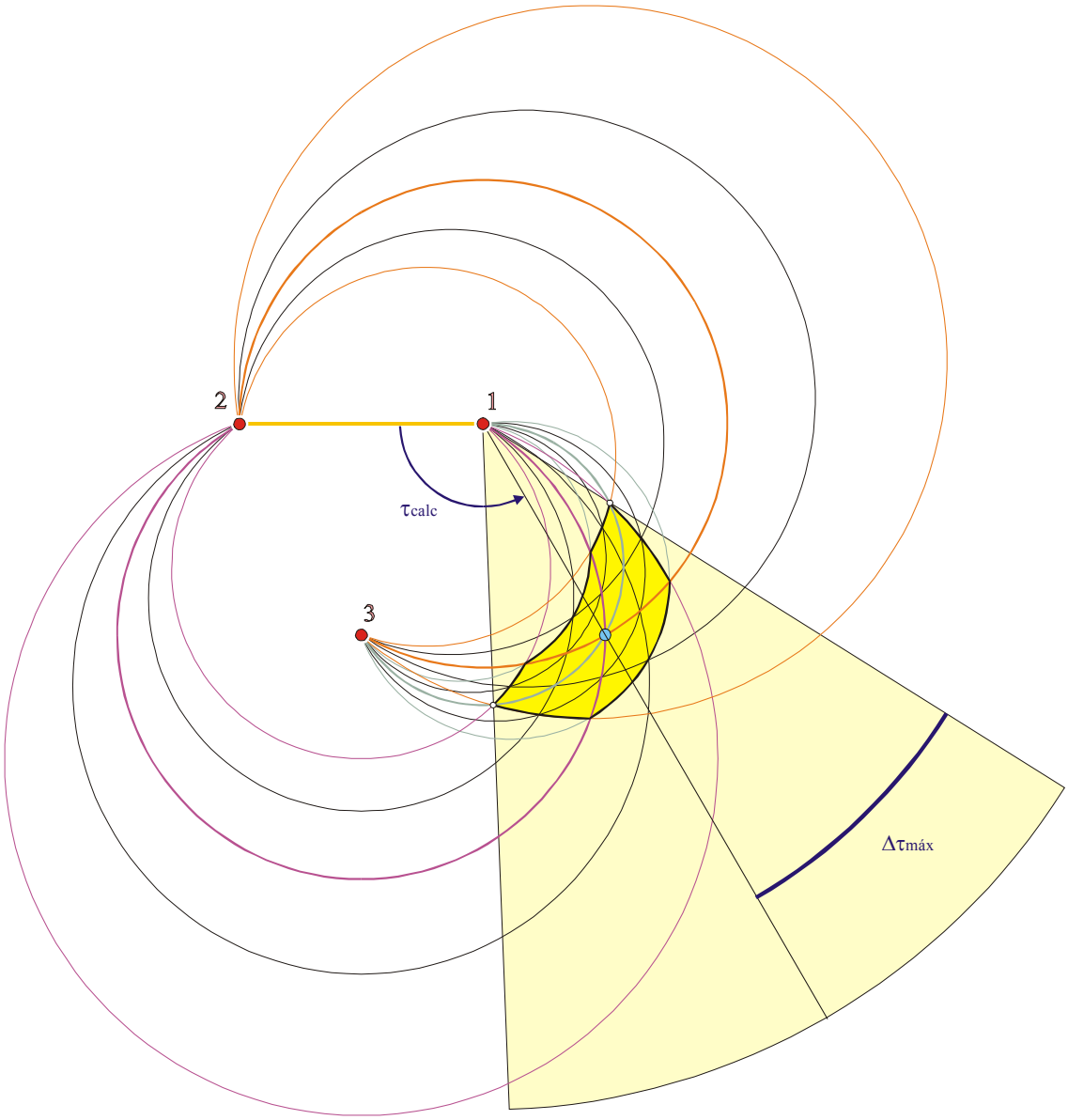


Figura 5.45: $\Delta\tau_{máx}$ numa superfície de incerteza de posição que resulta de λ_{12m} e λ_{31m} serem calculados a partir de medições independentes de λ_1, λ_2 e λ_3. Neste exemplo há dois pontos da superfície de incerteza de posição em que $\Delta\tau = \Delta\tau_{máx}$.

A determinação das coordenadas de $P_{\Delta\tau máx}$ no referencial x0y definido no plano de navegação permite o cálculo de $\Delta\tau_{max}$, uma vez que as coordenadas de P_{Rcalc} já estão calculadas e as da baliza 1 são conhecidas *a priori*. Os próximos parágrafos estão dedicados a esta tarefa.

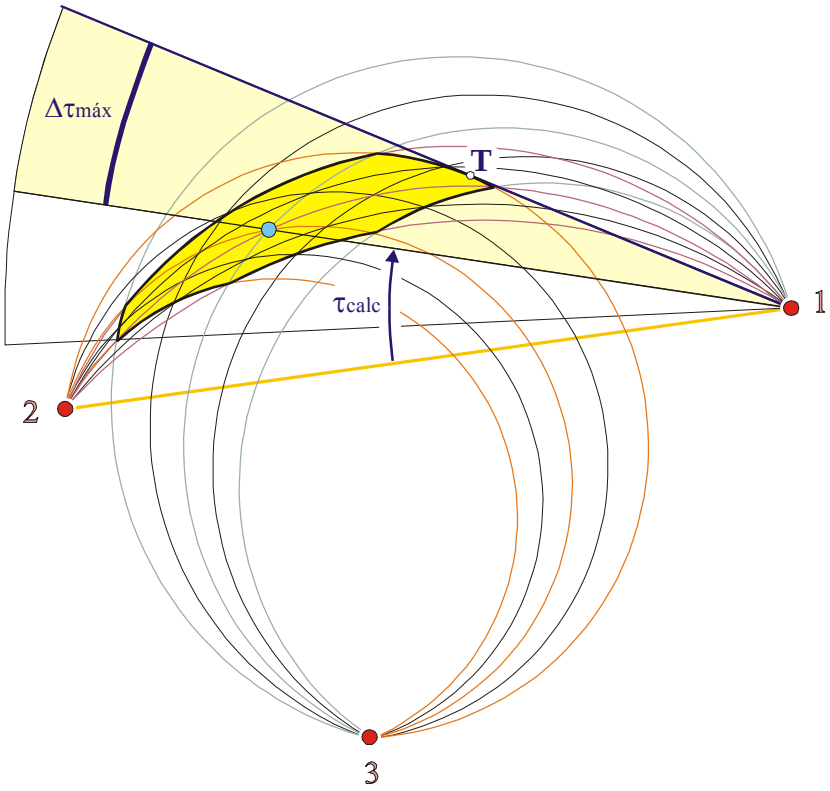


Figura 5.46: $\Delta\tau_{máx}$ numa superfície de incerteza de posição que resulta de λ_{12m} e λ_{31m} serem calculados a partir de medições independentes de λ_1, λ_2 e λ_3. Neste exemplo, o ponto da superfície de incerteza de posição em que $\Delta\tau = \Delta\tau_{máx}$ não é um vértice dessa superfície.

$P_{\Delta\tau máx}$ situa-se no contorno da superfície de incerteza de posição e, além disso, a recta que passa por $P_{\Delta\tau máx}$ e a baliza 1 não contém nenhum outro ponto dessa superfície. Há então duas situações a investigar:

1. $P_{\Delta\tau máx}$ é um vértice da superfície de incerteza de posição (Figura 5.44 e Figura 5.45);

2. $P_{\Delta\tau máx}$ não é um vértice da superfície de incerteza de posição, mas pertence a um arco de circunferência cujas extremidades se situam em vértices da superfície. A recta que passa por $P_{\Delta\tau máx}$ e a baliza 1 é tangente a esse arco (Figura 5.46).

Se λ_{12m} e λ_{31m} resultarem de medições independentes de λ_{12} e λ_{31}, a superfície de incerteza de posição é um quadrilátero curvilíneo cujos lados são arcos pertencentes a circunferências que passam pela baliza 1. Qualquer recta que passe pela baliza 1 não

pode ser tangente a nenhuma dessas circunferências a não ser na própria baliza 1. Por isso, $P_{\Delta\tau máx}$ tem de ser um vértice da superfície de incerteza de posição.

Se λ_{12m} e λ_{31m} forem calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, quatro dos seis lados da superfície de incerteza de posição são arcos pertencentes a circunferências que passam pela baliza 1. No entanto, os outros dois lados, Π_{16} e Π_{34} (Figura 5.37), são arcos pertencentes a circunferências que passam pelas balizas 2 e 3 mas, à excepção de uma delas, não passam pela baliza 1. Sempre que esta baliza se situa no exterior de uma circunferência, há duas rectas que se intersectam na baliza e são tangentes à circunferência (Figura 5.47). É então possível que $P_{\Delta\tau máx}$ seja um dos pontos de tangência e não um vértice da superfície de incerteza de posição.

Seja Π um dos arcos Π_{16} ou Π_{34}. A observação da Figura 5.48, da Figura 5.49 e da Figura 5.50 permite concluir que uma recta que passe pela baliza 1 só pode ser tangente a Π nas seguintes circunstâncias:

- Se as três balizas forem não colineares e o arco Π intersectar uma das regiões identificadas como E, F, G e H na Figura 5.48 (balizas ordenadas no sentido directo) e na Figura 5.49 (balizas ordenadas no sentido inverso).

  A estas regiões correspondem, no referencial $\lambda_{12}0\lambda_{31}$, valores de λ_{12} e de λ_{31} que estão contidos nos seguintes intervalos (Figura 5.22 e Figura 5.23):

  - para balizas ordenadas no sentido directo ($0º < \sigma < 180º$):

    E e F: $360º < \lambda_{12}+\lambda_{31} < 540º$ $\quad [360º+2\Delta < \lambda_{12m}+\lambda_{31m} < 540º-2\Delta]$

    G: $180º < \lambda_{31} < (\delta+180º)$ $\quad [180º-2\Delta < \lambda_{31m} < (\delta+180º)-2\Delta]$ (5.13)

    H: $180º < \lambda_{12} < (\sigma-\delta+180º)$ $\quad [180º-2\Delta < \lambda_{12m} < (\sigma-\delta+180º)-2\Delta]$

  - para balizas ordenadas no sentido inverso ($-180º < \sigma < 0º$):

    E e F: $180º < \lambda_{12}+\lambda_{31} < 360º$ $\quad [180º+2\Delta < \lambda_{12m}+\lambda_{31m} < 360º-2\Delta]$

    G: $(\delta+180º) < \lambda_{31} < 180º$ $\quad [(\delta+180º)+2\Delta < \lambda_{31m} < 180º+2\Delta]$ (5.14)

    H: $(\sigma-\delta+180º) < \lambda_{12} < 180º$ $\quad [(\sigma-\delta+180º)+2\Delta < \lambda_{12m} < 180º+2\Delta]$

  Se λ_{12m} e λ_{31m} pertencerem aos intervalos indicados para cada região, as superfícies de incerteza de posição que lhes estão associadas são delimitadas por arcos dos quais pelo menos um intersecta alguma dessas regiões.

- Se as três balizas forem colineares e a baliza 1 não for a baliza central (Figura 5.50) ou seja, se $\sigma = 180º$.

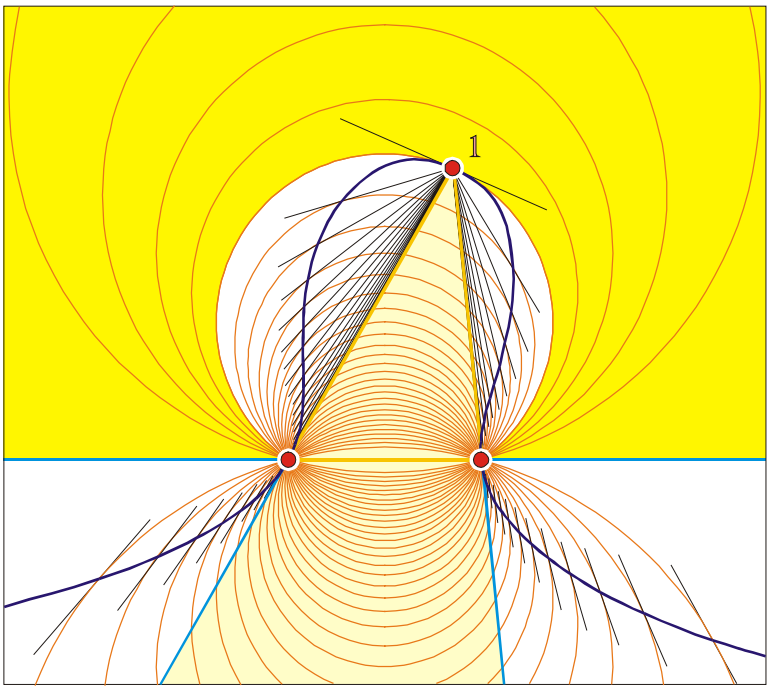


Figura 5.47: Sempre que a baliza 1 se situa no exterior de uma circunferência, há duas rectas que se intersectam na baliza e são tangentes à circunferência. A curva a azul escuro é o lugar geométrico de todos os pontos de tangência, que ocorrem sempre fora das regiões sombreadas a amarelo. A baliza 1 é interior às circunferências nas quais estão contidos os arcos da região sombreada a amarelo mais escuro. Nenhuma das rectas tangentes a esses arcos passa pela baliza 1. Além disso, nenhuma recta que passe pela baliza 1 e intersecte o segmento de recta que une as outras duas balizas pode ser tangente a alguma das circunferências que passam por essas balizas. Por isso, não pode haver pontos de tangência na região sombreada a amarelo mais claro.

Assim, há três circunstâncias nas quais uma recta que passe pela baliza 1 não pode ser tangente a Π:

- Se as três balizas forem não colineares e a baliza 1 estiver no interior da circunferência que contém o arco Π (corresponde à região sombreada a amarelo mais escuro, na Figura 5.48 e na Figura 5.49);

- Se as três balizas forem não colineares e o arco Π estiver integralmente contido na região do plano na qual uma recta que passe pela baliza 1 intersecta o segmento de recta que une as outras duas balizas (isto ocorre na região sombreada a amarelo mais claro, na Figura 5.48 e na Figura 5.49);

- Se as três balizas forem colineares e a baliza central for a baliza 1 (Figura 5.51).

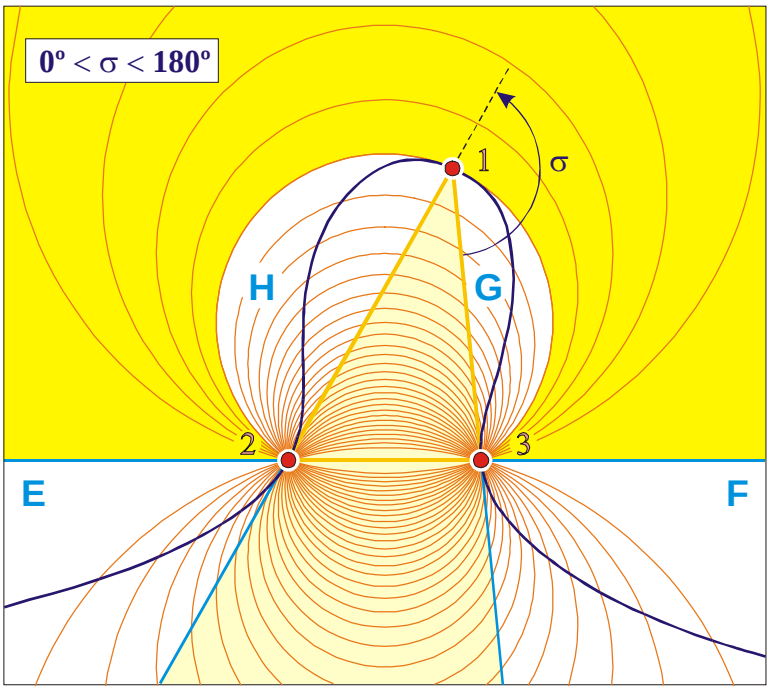


Figura 5.48: Três balizas não colineares ordenadas no sentido directo. Não há nenhuma recta que seja tangente a um dos arcos da região sombreada a amarelo ($\lambda_{23} < 180º-\sigma$) e, simultaneamente, passe pela baliza 1.

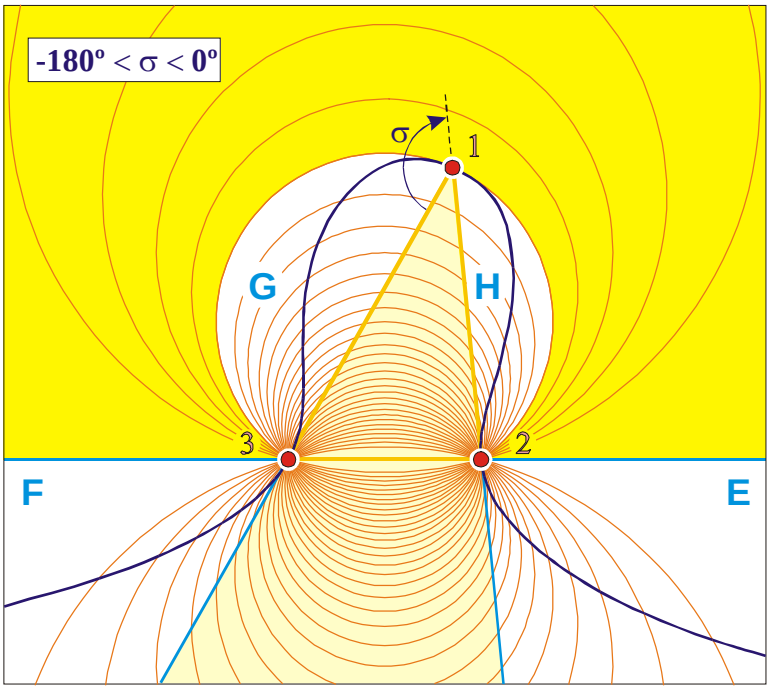


Figura 5.49: Três balizas não colineares ordenadas no sentido inverso. Não há nenhuma recta que seja tangente a um dos arcos da região sombreada a amarelo ($\lambda_{23} > 180º-\sigma$) e, simultaneamente, passe pela baliza 1.

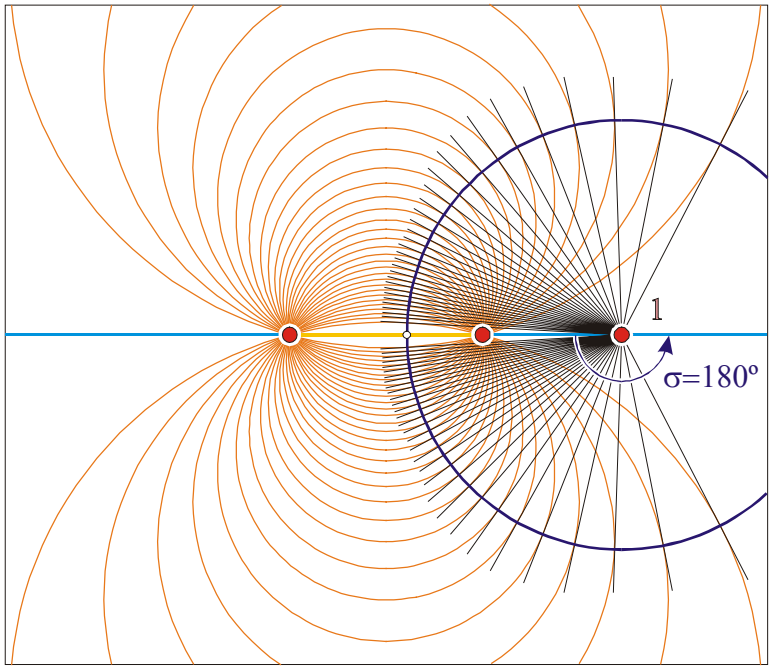


Figura 5.50: Três balizas colineares em que a baliza 1 não é a baliza central. Para cada arco de circunferência há uma recta que lhe é tangente e passa pela baliza 1.

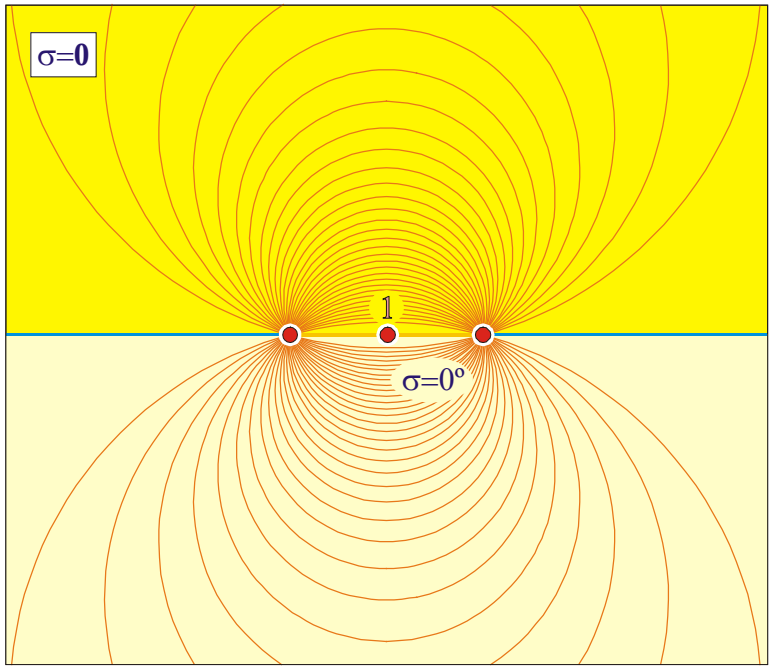
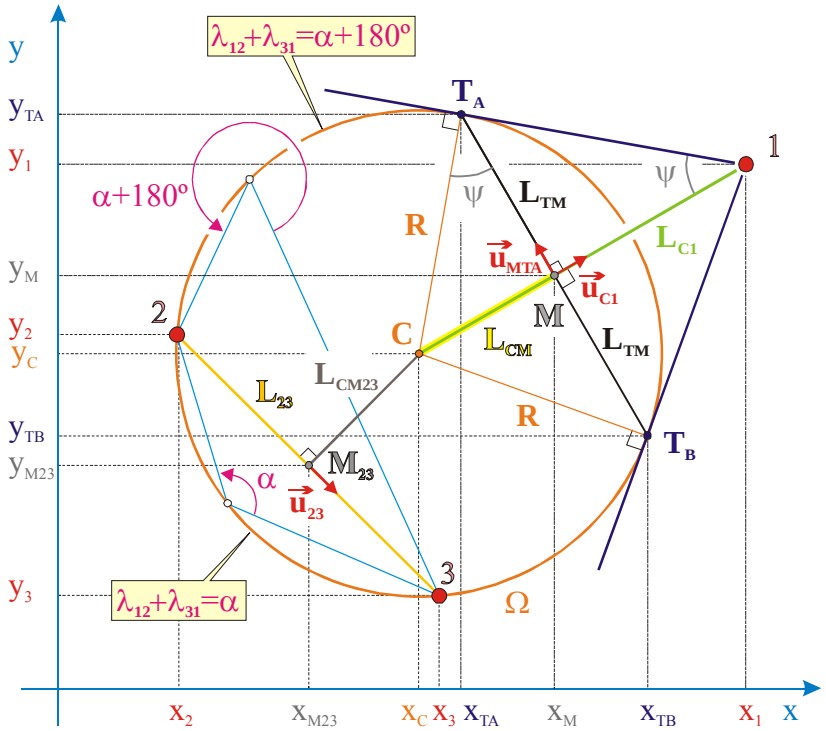


Figura 5.51: Três balizas colineares em que a baliza central é a baliza 1. Não há nenhuma recta que seja tangente a um dos arcos e, simultaneamente, passe pela baliza 1.

O algoritmo descrito na Figura 5.52 serve para determinar as coordenadas dos pontos T_A e T_B nos quais duas rectas que se intersectam na baliza 1 são tangentes à circunferência Ω sobre a qual se encontra o arco Π. À partida, cada um desses pontos pode ou não pertencer a Π, pelo que tal pertença tem de ser verificada depois de o algoritmo ser executado.



x_1 , y_1	-	Coordenadas da baliza **1**
x_2 , y_2	-	Coordenadas da baliza **2**
x_3 , y_3	-	Coordenadas da baliza **3**
R	-	Raio da circunferência **Ω**, que contém o arco **Π**
L_{23}	-	Distância entre as balizas **2** e **3**
λ_Π	-	Valor de $\lambda_{12}+\lambda_{31}$ ao longo do arco **Π**
C	-	Centro da circunferência **Ω** Coordenadas: x_C , y_C
L_{C1}	-	Distância entre o ponto **C** e a baliza **1**
L_{TM}	-	Distância entre um dos pontos $\mathbf{T_A}$ ou $\mathbf{T_B}$ e o ponto **M**
L_{CM}	-	Distância entre os pontos **C** e **M**
L_{CM23}	-	Quantidade (pode ser negativa) cujo valor absoluto é a distância entre os pontos **C** e $\mathbf{M_{23}}$
$\vec{u}_{C1}$	-	Vector unitário orientado de **C** para a baliza **1** Componentes: u_{C1x} , u_{C1y}
$\vec{u}_{MTA}$	-	Vector unitário orientado de **M** para $\mathbf{T_A}$ Componentes: u_{MTAx} , u_{MTAy}
$\vec{u}_{23}$	-	Vector unitário orientado da baliza **2** para a baliza **3** Componentes: u_{M23Cx} , u_{M23Cy}
M	-	Ponto médio do segmento que une os pontos $\mathbf{T_A}$ e $\mathbf{T_B}$ Coordenadas: x_M , y_M
M_{23}	-	Ponto médio do segmento que une as balizas **A** e **B** Coordenadas: x_{M23} , y_{M23}
x_{TA}, y_{TA}	-	Coordenadas do ponto $\mathbf{T_A}$
x_{TB}, y_{TB}	-	Coordenadas do ponto $\mathbf{T_B}$

1. $R = \left| \dfrac{L_{23}}{2 \cdot \operatorname{sen} \lambda_\Pi} \right|$ (Anexo G)
2. $L_{CM23} = -\dfrac{L_{23}}{2 \cdot \operatorname{tg} \lambda_\Pi}$ (Anexo G)
3. $u_{23x} = \dfrac{x_3 - x_2}{L_{23}}$; $u_{23y} = \dfrac{y_3 - y_2}{L_{23}}$
4. $x_C = x_{M23} - L_{CM23} \cdot u_{23y}$
5. $y_C = y_{M23} + L_{CM23} \cdot u_{23x}$
6. $L_{C1} = \sqrt{(x_1 - x_C)^2 + (y_1 - y_C)^2}$
7. $\psi = \operatorname{arcsen} \dfrac{R}{L_{C1}}$
8. $L_{CM} = R \cdot \operatorname{sen} \psi$
9. $L_{TM} = R \cdot \cos \psi$
10. $u_{C1x} = \dfrac{x_1 - x_C}{L_{C1}}$; $u_{C1y} = \dfrac{y_1 - y_C}{L_{C1}}$
11. $u_{MTAx} = -u_{C1y}$; $u_{MTAy} = u_{C1x}$
12. $x_M = x_C + L_{CM} \cdot u_{C1x}$
13. $y_M = y_C + L_{CM} \cdot u_{C1y}$
14. $x_{TA} = x_M + L_{TM} \cdot u_{MTAx}$
15. $y_{TA} = y_M + L_{TM} \cdot u_{MTAy}$
16. $x_{TB} = x_M - L_{TM} \cdot u_{MTAx}$
17. $y_{TB} = y_M - L_{TM} \cdot u_{MTAy}$

Figura 5.52: Cálculo das coordenadas de T_A e T_B.

No algoritmo da Figura 5.52, Ω é uma circunferência que contém as balizas 2 e 3 mas não a baliza 1 e atravessa pelo menos uma das regiões do plano de navegação nas quais uma recta que passa pela baliza 1 pode ser tangente a Ω. Ao longo desta circunferência podem ocorrer até três valores diferentes de $\lambda_{12}+\lambda_{31}$:

- Com três balizas não colineares, ordenadas quer no sentido directo (Figura 5.53) quer no sentido inverso (Figura 5.54), o ângulo $\lambda_{12}+\lambda_{31}$ pode assumir dois ou três valores diferentes ao longo de Ω. As diferenças entre valores distintos de $\lambda_{12}+\lambda_{31}$ obtidos em cada arco da mesma circunferência são múltiplos de 180º.

- Com três balizas colineares em que a baliza 3 é a baliza central (Figura 5.55) ou a baliza 2 é a baliza central (Figura 5.56), o ângulo $\lambda_{12}+\lambda_{31}$ assume dois valores diferentes ao longo Ω. A diferença entre os dois valores de $\lambda_{12}+\lambda_{31}$ obtidos em cada arco da mesma circunferência é de 180º.

Uma vez que, para todas as configurações de balizas que foram referidas, as diferenças entre valores distintos de $\lambda_{12}+\lambda_{31}$ obtidos em cada arco da mesma circunferência são múltiplos de 180º, os valores de R e L_{CM23}, calculados nas linhas 1 e 2 do algoritmo, são sempre os mesmos para cada circunferência, independentemente do valor de $\lambda_{12}+\lambda_{31}$ utilizado no seu cálculo. Por exemplo, na situação representada no gráfico que acompanha o algoritmo (Figura 5.52) pode utilizar-se indiferentemente um dos ângulos $\lambda_{12}+\lambda_{31}=\alpha$ ou $\lambda_{12}+\lambda_{31}=\alpha+180º$. Na prática, o que se pretende é que Ω contenha o arco Π, que é um dos arcos Π_{16} ou Π_{34}. Seja λ_{Π} o valor de $\lambda_{12}+\lambda_{31}$ que ocorre em algum ponto de Π:

- Para que a circunferência Ω contenha o arco Π_{16}, os valores de R e L_{CM23} podem ser calculados utilizando $\lambda_{\Pi} = \lambda_{12m}+\lambda_{31m}-2\Delta\lambda$;

- Para que a circunferência Ω contenha o arco Π_{34}, os valores de R e L_{CM23} podem ser calculados utilizando $\lambda_{\Pi} = \lambda_{12m}+\lambda_{31m}+2\Delta\lambda$.

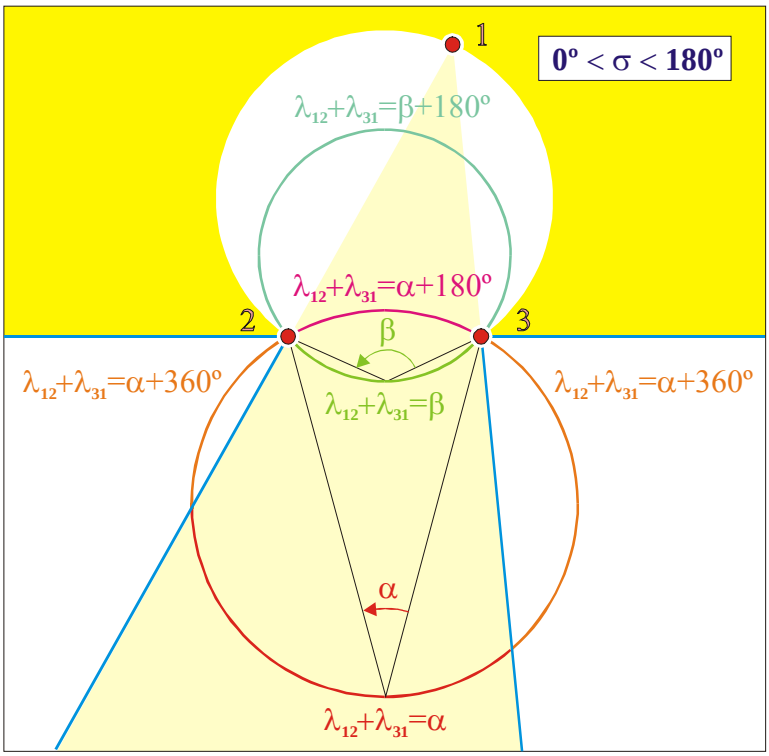


Figura 5.53: Com três balizas não colineares ordenadas no sentido directo o ângulo $\lambda_{12}+\lambda_{31}$ pode assumir dois ou três valores diferentes ao longo de uma circunferência que contém as balizas 1 e 2 mas não a baliza 1 e atravessa pelo menos uma das regiões nas quais uma recta que passa pela baliza 1 pode ser tangente a essa circunferência. As diferenças entre valores distintos de $\lambda_{12}+\lambda_{31}$ obtidos em cada arco da mesma circunferência são múltiplos de 180º.

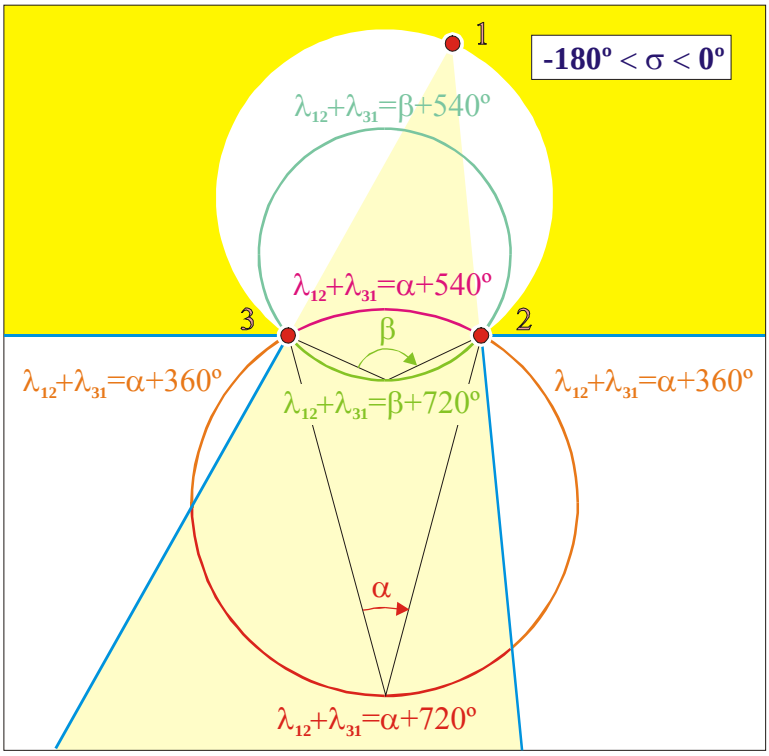


Figura 5.54: Com três balizas não colineares ordenadas no sentido inverso o ângulo $\lambda_{12}+\lambda_{31}$ pode assumir dois ou três valores diferentes ao longo de uma circunferência que contém as balizas 1 e 2 mas não a baliza 1 e atravessa pelo menos uma das regiões nas quais uma recta que passa pela baliza 1 pode ser tangente a essa circunferência. As diferenças entre valores distintos de $\lambda_{12}+\lambda_{31}$ obtidos em cada arco da mesma circunferência são múltiplos de 180º.

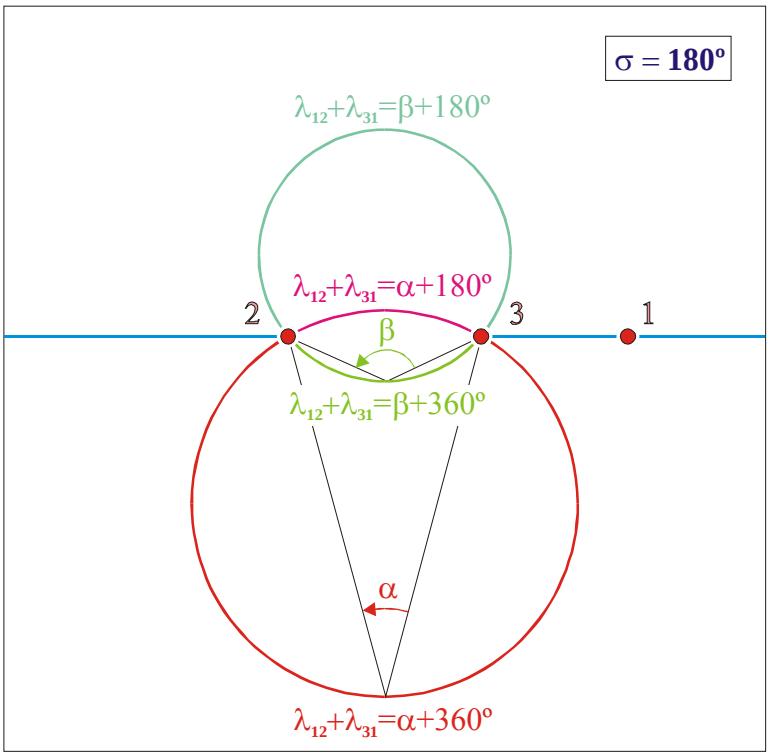


Figura 5.55: Com três balizas colineares em que a baliza 3 é a baliza central o ângulo $\lambda_{12}+\lambda_{31}$ assume dois valores diferentes ao longo de uma circunferência que contém as balizas 1 e 2 mas não a baliza 1. A diferença entre os dois valores de $\lambda_{12}+\lambda_{31}$ obtidos em cada arco da mesma circunferência é de 180º.

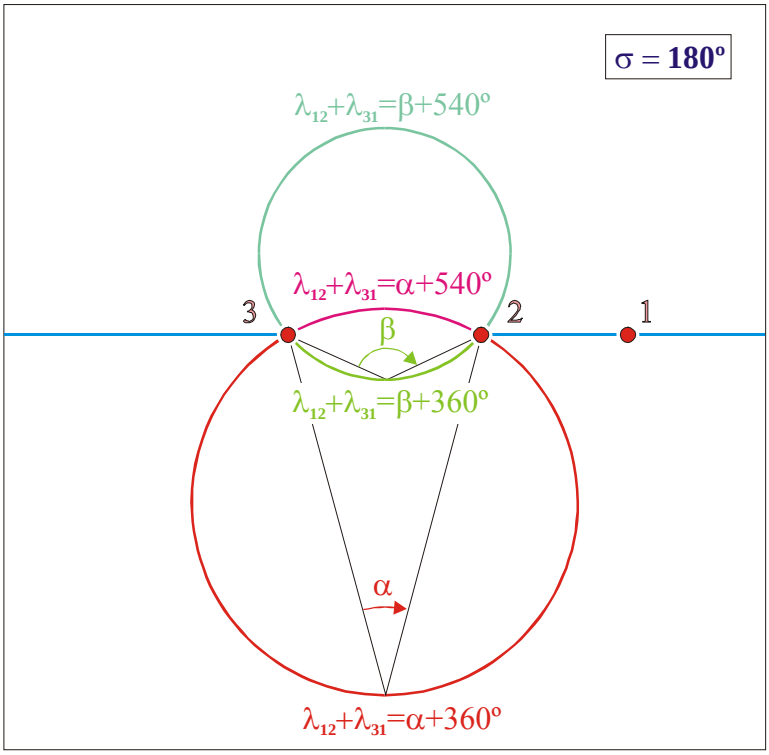


Figura 5.56: Com três balizas colineares em que a baliza 2 é a baliza central o ângulo $\lambda_{12}+\lambda_{31}$ assume dois valores diferentes ao longo de uma circunferência que contém as balizas 1 e 2 mas não a baliza 1. A diferença entre os dois valores de $\lambda_{12}+\lambda_{31}$ obtidos em cada arco da mesma circunferência é de 180º.

Depois de calcular as coordenadas de T_A e T_B, é necessário averiguar se cada um destes pontos pertence ao arco Π. Isso pode ser feito verificando se os valores de λ_{12} e λ_{31} calculados nos pontos T_A e T_B tornam verdadeiras as três condições que estabelecem a pertença de um ponto ao arco Π:

- Em todos os pontos do arco Π_{16}, e só nesses pontos, verificam-se simultaneamente as seguintes condições (Figura 5.37):

    1. $\lambda_{12}+\lambda_{31} = \lambda_{12m}+\lambda_{31m}-2\Delta\lambda$
    2. $\lambda_{12m}-2\Delta\lambda \leq \lambda_{12} \leq \lambda_{12m}$
    3. $\lambda_{31m}-2\Delta\lambda \leq \lambda_{31} \leq \lambda_{31m}$

- Em todos os pontos do arco Π_{34}, e só nesses pontos, verificam-se simultaneamente as seguintes condições (Figura 5.37):

    1. $\lambda_{12}+\lambda_{31} = \lambda_{12m}+\lambda_{31m}+2\Delta\lambda$
    2. $\lambda_{12m} \leq \lambda_{12} \leq \lambda_{12m}+2\Delta\lambda$
    3. $\lambda_{31m} \leq \lambda_{31} \leq \lambda_{31m}+2\Delta\lambda$

Este processo pode ser substancialmente simplificado. De facto, a condição 1 está, para cada arco, garantida *a priori* uma vez que o respectivo valor de $\lambda_{12}+\lambda_{31}$ é uma das entradas do algoritmo da Figura 5.52. Além disso, como se demonstra nos próximos parágrafos, é suficiente que se verifique a condição 2 (poderia ser a condição 3 em vez da condição 2).

Seja Π_{12} um arco de circunferência definido por um valor de λ_{12} e cujas extremidades são as balizas 1 e 2. Como Ω é uma circunferência que contém as balizas 2 e 3 mas não a baliza 1 (Figura 5.52), é sempre verdade que:

- O arco Π_{12} e a circunferência Ω têm na baliza 2 um ponto comum. Por isso só podem, no máximo, intersectar-se em mais um ponto[28];

- A cada arco Π_{12} corresponde um valor único de λ_{12} que o distingue de todos os outros arcos da mesma família.

Quer isto dizer que não pode haver dois pontos de Ω nos quais ocorra o mesmo valor de λ_{12}. A cada ponto de Ω corresponde apenas um valor de λ_{12} que é diferente dos valores de λ_{12} que ocorrem nos outros pontos dessa circunferência. Nos pontos do arco Π, contido em Ω, verifica-se que $\lambda_{12mín} \leq \lambda_{12} \leq \lambda_{12máx}$. De acordo com o exposto, para que um ponto P de Ω pertença também a Π é condição necessária e suficiente que o valor de λ_{12} que ocorre em P pertença ao intervalo $[\lambda_{12mín}, \lambda_{12máx}]$. Assim sendo, verifica-se que:

- Para que o ponto T (pode ser T_A ou T_B) da circunferência Ω pertença também ao arco Π_{16}, é necessário e suficiente que $\lambda_{12m}-2\Delta\lambda \leq \lambda_{12T} \leq \lambda_{12m}$, em que λ_{12T} é o valor de λ_{12} calculado nesse ponto;

- Para que o ponto T (pode ser T_A ou T_B) da circunferência Ω pertença também ao arco Π_{34}, é necessário e suficiente que $\lambda_{12m} \leq \lambda_{12T} \leq \lambda_{12m}+2\Delta\lambda$, em que λ_{12T} é o valor de λ_{12} calculado nesse ponto.

Pelo menos em princípio, é possível que os pontos T_A e T_B pertencentes à circunferência Ω pertençam ambos ao arco Π (Figura 5.57). Nesse caso, o maior $\Delta\tau$ verificado no arco ocorre seguramente num desses dois pontos. Quando apenas um dos pontos T_A ou T_B pertence a Π, o maior $\Delta\tau$ verificado no arco pode ocorrer nesse ponto (Figura 5.58 e Figura 5.59) mas é também possível que ocorra numa das suas extremidades (Figura 5.60), que são vértices da superfície de incerteza de posição. Finalmente, pode suceder que nenhum dos pontos T_A e T_B pertença a Π (Figura 5.61).

[28] Se Ω contivesse a baliza 1 e o robô se encontrasse sobre esta circunferência, então todos os pontos de Π_{12} pertenceriam a Ω.

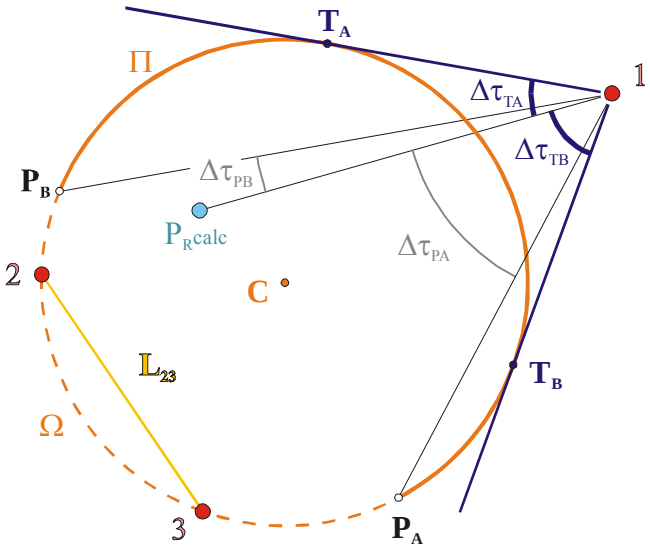


Figura 5.57: Os pontos T_A e T_B pertencem ambos ao arco Π e o maior Δτ verificado no arco é $\Delta\tau_{TB}$.

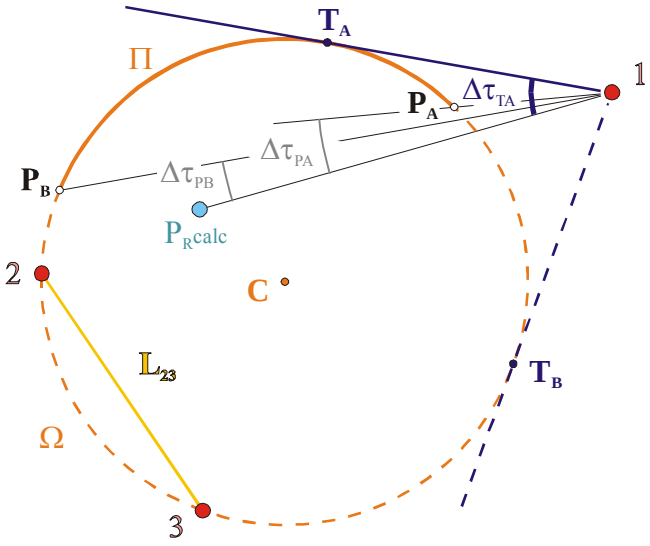


Figura 5.58: O ponto T_B não pertence ao arco Π e o maior Δτ verificado no arco é $\Delta\tau_{TA}$.

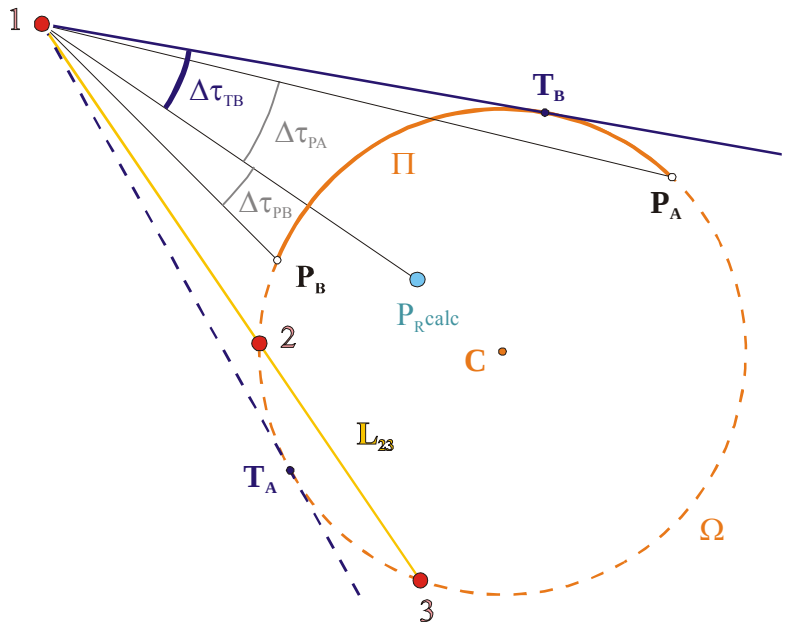


Figura 5.59: O ponto T_A não pertence ao arco Π e o maior $\Delta\tau$ verificado no arco é $\Delta\tau_{TB}$.

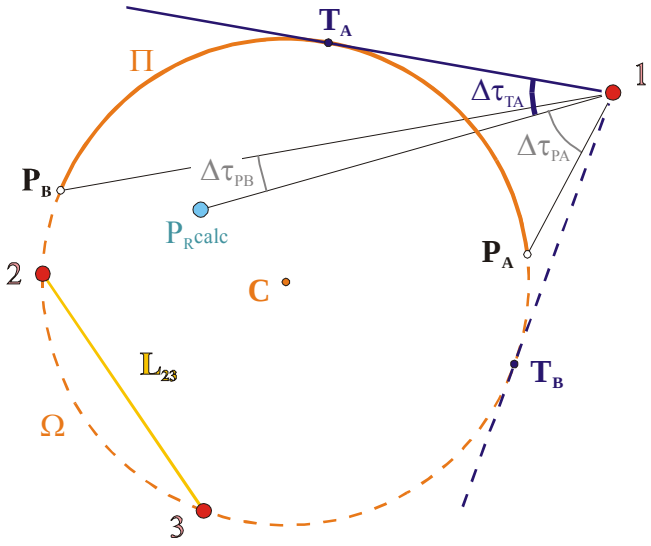


Figura 5.60: O ponto T_B não pertence ao arco Π e o maior $\Delta\tau$ verificado no arco é $\Delta\tau_{PA}$.

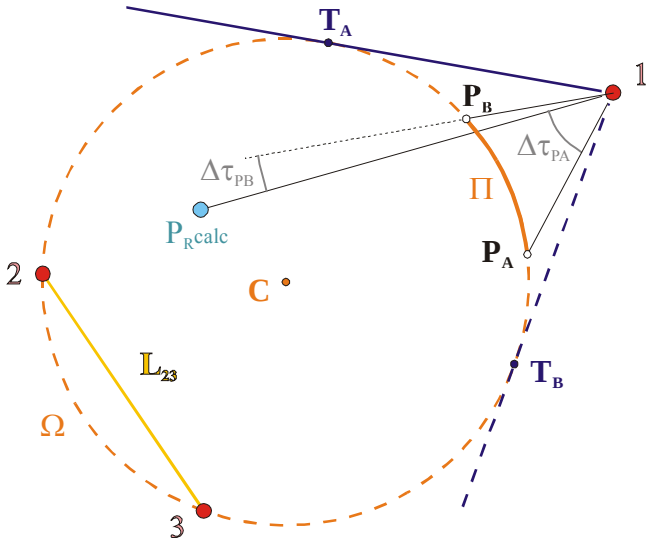


Figura 5.61: Nenhum dos pontos T_A e T_B pertence ao arco Π e o maior Δτ verificado no arco é $\Delta\tau_{PA}$.

Em suma, este é o algoritmo sugerido para determinar o erro máximo de orientação $\Delta\theta_{Rmáx}$ (o passo 3 deve ser omitido se λ_{12m} e λ_{31m} resultarem de medições independentes de λ_{12} e λ_{31}):

1. Fazer $\Delta\tau_{máx} = 0$;
2. Em cada vértice da superfície de incerteza de posição:
   a. determinar o valor de Δτ, a partir das coordenadas – previamente calculadas – do vértice e de P_{Rcalc} no referencial x0y;
   b. se $\Delta\tau > \Delta\tau_{máx}$ então $\Delta\tau_{máx} = \Delta\tau$;
3. Se for possível que $\Delta\tau_{máx}$ não ocorra num vértice da superfície de incerteza de posição, ou seja, se

$$\left(0^\circ<\sigma<180^\circ \wedge \begin{bmatrix} 180^\circ-2\Delta\lambda<\lambda_{12m}<(\sigma-\delta+180^\circ)-2\Delta\lambda \\ \vee\ 180^\circ-2\Delta\lambda<\lambda_{31m}<(\delta+180^\circ)-2\Delta\lambda \\ \vee\ 360^\circ+2\Delta\lambda<\lambda_{12m}+\lambda_{31m}<540^\circ-2\Delta\lambda \end{bmatrix}\right)$$

$$\vee$$

$$\left(-180^\circ<\sigma<0^\circ \wedge \begin{bmatrix} (\sigma-\delta+180^\circ)+2\Delta\lambda<\lambda_{12m}<180^\circ+2\Delta\lambda \\ \vee\ (\delta+180^\circ)+2\Delta\lambda<\lambda_{31m}<180^\circ+2\Delta\lambda \\ \vee\ 180^\circ+2\Delta\lambda<\lambda_{12m}+\lambda_{31m}<360^\circ-2\Delta\lambda \end{bmatrix}\right)$$

$$\vee$$

$$\sigma = 180^\circ$$

então

a. fazer $\lambda_{\Pi} = \lambda_{12m}+\lambda_{31m}-2\Delta\lambda$ $(\Pi=\Pi_{16})$;

b. determinar as coordenadas dos pontos T_A e T_B (algoritmo da Figura 5.52);

c. seja λ_{12TA} o valor de λ_{12} no ponto T_A, calculado a partir das coordenadas desse ponto e das coordenadas das balizas 1 e 2 no referencial x0y;

d. se $\lambda_{12m}-2\Delta\lambda \leq \lambda_{12TA} \leq \lambda_{12m}$ (ou seja, se T_A pertence ao arco Π_{16}) então

   i. determinar o valor de $\Delta\tau_{TA}$, a partir das coordenadas de T_A e de P_{Rcalc} no referencial x0y;

   ii. se $\Delta\tau_{TA} > \Delta\tau_{máx}$ então $\Delta\tau_{máx} = \Delta\tau_{TA}$;

e. seja λ_{12TB} o valor de λ_{12} no ponto T_B, calculado a partir das coordenadas desse ponto e das coordenadas das balizas 1 e 2 no referencial x0y;

f. se $\lambda_{12m}-2\Delta\lambda \leq \lambda_{12TB} \leq \lambda_{12m}$ (ou seja, se T_B pertence ao arco Π_{16}) então

   i. determinar o valor de $\Delta\tau_{TB}$, a partir das coordenadas de T_B e de P_{Rcalc} no referencial x0y;

   ii. se $\Delta\tau_{TB} > \Delta\tau_{máx}$ então $\Delta\tau_{máx} = \Delta\tau_{TB}$;

g. fazer $\lambda_{12\Pi} = \lambda_{12m}+\lambda_{31m}+2\Delta\lambda$ $(\Pi=\Pi_{34})$;

h. determinar as coordenadas dos pontos T_A e T_B (algoritmo da Figura 5.52);

i. seja λ_{12TA} o valor de λ_{12} no ponto T_A, calculado a partir das coordenadas desse ponto e das coordenadas das balizas 1 e 2 no referencial x0y;

j. se $\lambda_{12m} \leq \lambda_{12TA} \leq \lambda_{12m}+2\Delta\lambda$ (ou seja, se T_A pertence ao arco Π_{34}) então

   i. determinar o valor de $\Delta\tau_{TA}$, a partir das coordenadas de T_A e de P_{Rcalc} no referencial x0y;

   ii. se $\Delta\tau_{TA} > \Delta\tau_{máx}$ então $\Delta\tau_{máx} = \Delta\tau_{TA}$;

k. seja λ_{12TB} o valor de λ_{12} no ponto T_B, calculado a partir das coordenadas desse ponto e das coordenadas das balizas 1 e 2 no referencial x0y;

l. se $\lambda_{12m} \leq \lambda_{12TB} \leq \lambda_{12m}+2\Delta\lambda$ (ou seja, se T_B pertence ao arco Π_{34}) então

   i. determinar o valor de $\Delta\tau_{TB}$, a partir das coordenadas de T_B e de P_{Rcalc} no referencial x0y;

   ii. se $\Delta\tau_{TB} > \Delta\tau_{máx}$ então $\Delta\tau_{máx} = \Delta\tau_{TB}$;

4. $\Delta\theta_{Rmáx} = \Delta\tau_{máx} + \Delta\lambda$;

Este algoritmo possui as seguintes características, que são comuns ao algoritmo sugerido para cálculo do erro máximo de posição:

a) Funciona com todos os algoritmos de autolocalização absoluta por triangulação com três balizas (é independente do algoritmo de triangulação utilizado).

b) É exacto (não há aproximações que lhe sejam inerentes).

c) Por ser exacto, não requer que a incerteza de medição de ângulos se mantenha abaixo de um determinado limiar.

d) Não requer o cálculo de derivadas parciais difíceis de obter.

e) Pode utilizar-se quando λ_{12m} e λ_{31m} resultam de medições independentes dos ângulos λ_{12} e λ_{31} ou quando são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3.

f) Só deve ser utilizado se o algoritmo de triangulação usado no cálculo da posição for suficientemente rápido uma vez que, para cada posição calculada do robô, esse algoritmo tem de ser executado

   - 5 vezes se λ_{12m} e λ_{31m} resultarem de medições independentes dos ângulos λ_{12} e λ_{31};

   - 7 vezes se λ_{12m} e λ_{31m} forem calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3).

5.6.4 Redução da Superfície Navegável

O correcto funcionamento do método de cálculo dos erros máximos de posição e de orientação requer que, para cada posição calculada, todos os vértices da respectiva superfície de incerteza de medição possuam imagens no plano de navegação e que nenhuma dessas imagens coincida com a posição de uma das três balizas utilizadas. Isto exclui os pontos situados em algumas zonas do plano de navegação, nas quais não é possível estimar as incertezas associadas à posição e à orientação calculadas. Estas são inúteis para efeitos de navegação quando não se fazem acompanhar das respectivas incertezas. A redução da superfície navegável é tanto maior quanto maior for a incerteza de medição de ângulos.

Por definição, λ_{12} e λ_{31} são inferiores a 360º e superiores ou iguais a 0º. Este intervalo de valores é suficiente para descrever todos os ângulos que podem formar entre si os segmentos de recta que unem o robô a cada baliza num dado instante.

No entanto, nos pontos do plano de navegação em que λ_{12m} ou λ_{31m} são próximos de 0º ou 360º, as coordenadas de alguns vértices das superfícies de incerteza de medição podem assumir valores inferiores a 0º ou superiores ou iguais a 360º. Para efeitos de localização do robô não há qualquer inconveniente em considerar esses valores nos cálculos. De facto, se a posição de um ponto P do plano de navegação ficar definida pelo par de ângulos λ_{12P} e λ_{31P}, então a posição desse ponto fica igualmente definida por qualquer par de ângulos λ'_{12P} e λ'_{31P} gerados pelas expressões

$$\begin{aligned} \lambda'_{12P} &= \lambda_{12P} \pm i \cdot 360º \qquad i = 0,1,2,... \\ \lambda'_{31P} &= \lambda_{31P} \pm j \cdot 360º \qquad j = 0,1,2,... \end{aligned} \tag{5.15}$$

Na Figura 5.62 pode ver-se a representação no referencial $\lambda_{12}0\lambda_{31}$ dos pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação para valores de λ_{12} e λ_{31} compreendidos entre –360º e 720º, com três balizas não colineares ordenadas no sentido directo. No centro da figura está o gráfico correspondente a λ_{12} e λ_{31} compreendidos entre 0º e 360º. À volta do gráfico central estão oito gráficos idênticos, obtidos por translação do primeiro. O processo é aplicável aos outros tipos de configurações de balizas. As regiões a branco são formadas por pontos que não possuem imagem no plano de navegação.

Seguidamente apresentam-se as condições que não se devem verificar, em cada configuração de balizas, a fim de garantir que todos os vértices de uma superfície de incerteza de medição se encontram fora dessas regiões e que nenhuma das imagens desses vértices coincide com a posição de uma das três balizas.

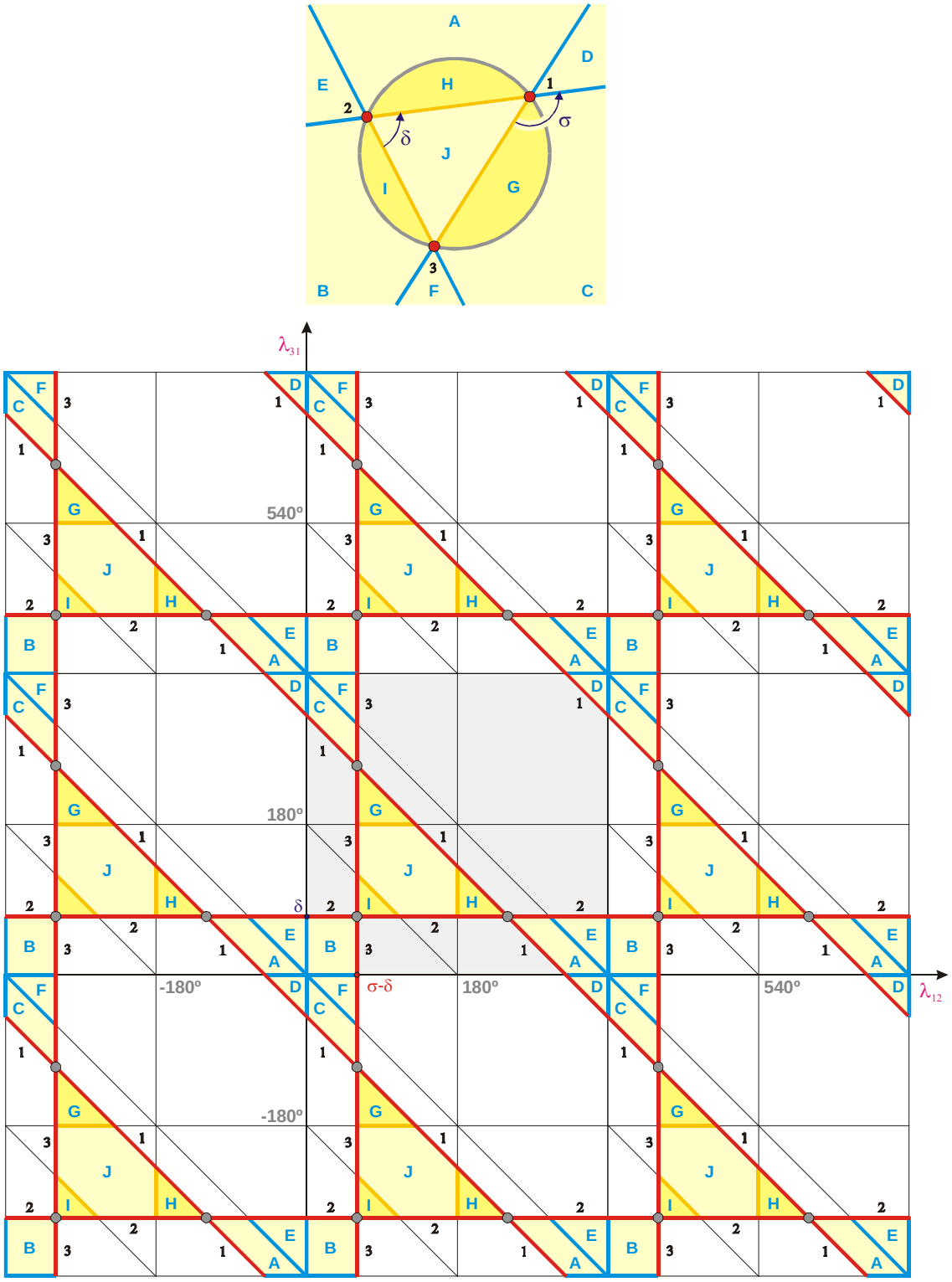


Figura 5.62: Representação no referencial $\lambda_{12}0\lambda_{31}$ dos pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação para valores de λ_{12} e λ_{31} compreendidos entre –360º e 720º, com três balizas não colineares ordenadas no sentido directo.

Se λ_{12m} e λ_{31m} resultarem de medições independentes de λ_{12} e λ_{31} realizadas com uma incerteza $\pm\Delta\lambda$, para garantir que todos os vértices da superfície de incerteza de medição possuem imagens no plano de navegação e que nenhuma dessas imagens coincide com a posição de uma das três balizas utilizadas, é necessário e suficiente que não se verifique nenhuma das seguintes condições:

- Para três balizas não colineares:
  - $\left|\lambda_{12m}-(\sigma-\delta)\right|\le\Delta\lambda$
  - $\left|\lambda_{12m}-(\sigma-\delta+360^\circ)\right|\le\Delta\lambda$
  - $\left|\lambda_{31m}-\delta\right|\le\Delta\lambda$ (5.16)
  - $\left|\lambda_{31m}-(\delta+360^\circ)\right|\le\Delta\lambda$
  - $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+180^\circ)\right|\le 2\Delta\lambda$
  - $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+540^\circ)\right|\le 2\Delta\lambda$
- Para três balizas colineares e a baliza 1 entre as balizas 2 e 3 (σ=0º):
  - $\lambda_{12m}\le\Delta\lambda$ [29]
  - $\lambda_{12m}\ge 360^\circ-\Delta\lambda$ [30]
  - $\lambda_{31m}\le\Delta\lambda$ [31] (5.17)
  - $\lambda_{31m}\ge 360^\circ-\Delta\lambda$ [32]
  - $\left|(\lambda_{12m}+\lambda_{31m})-180^\circ\right|\le 2\Delta\lambda$ [33]
  - $\left|(\lambda_{12m}+\lambda_{31m})-540^\circ\right|\le 2\Delta\lambda$ [34]
- Para três balizas colineares e a baliza 2 entre as balizas 1 e 3 (σ=180º e δ=180º):
  - $\lambda_{12m}\le\Delta\lambda$ [29]
  - $\lambda_{12m}\ge 360^\circ-\Delta\lambda$ [30] (5.18)
  - $\left|\lambda_{31m}-180^\circ\right|\le\Delta\lambda$ [31]
  - $\left|(\lambda_{12m}+\lambda_{31m})-360^\circ\right|\le 2\Delta\lambda$ [33]
- Para três balizas colineares e a baliza 3 entre as balizas 1 e 2 (σ=180º e δ=0º):
  - $\left|\lambda_{12m}-180^\circ\right|\le\Delta\lambda$ [29]
  - $\lambda_{31m}\le\Delta\lambda$ [31] (5.19)
  - $\lambda_{31m}\ge 360^\circ-\Delta\lambda$ [32]
  - $\left|(\lambda_{12m}+\lambda_{31m})-360^\circ\right|\le 2\Delta\lambda$ [33]

[29] É um caso particular de $\left|\lambda_{12m}-(\sigma-\delta)\right|\le\Delta\lambda$.

[30] É um caso particular de $\left|\lambda_{12m}-(\sigma-\delta+360^\circ)\right|\le\Delta\lambda$.

[31] É um caso particular de $\left|\lambda_{31m}-\delta\right|\le\Delta\lambda$.

[32] É um caso particular de $\left|\lambda_{31m}-(\delta+360^\circ)\right|\le\Delta\lambda$.

[33] É um caso particular de $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+180^\circ)\right|\le 2\Delta\lambda$

[34] É um caso particular de $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+540^\circ)\right|\le 2\Delta\lambda$

Assim, com qualquer configuração de balizas, para ser possível efectuar a autolocalização e determinar os erros máximos de posição e de orientação quando a posição calculada do robô no plano de navegação é a imagem do ponto (λ_{12m}, λ_{31m}) no referencial $\lambda_{12}0\lambda_{23}$, supondo que λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31} efectuadas com uma incerteza $\pm\Delta\lambda$, é necessário e suficiente que não se verifique nenhuma das seguintes condições:

- $\left|\lambda_{12m} - (\sigma - \delta)\right| \leq \Delta\lambda$
- $\left|\lambda_{12m} - (\sigma - \delta + 360^o)\right| \leq \Delta\lambda$
- $\left|\lambda_{31m} - \delta\right| \leq \Delta\lambda$ (5.20)
- $\left|\lambda_{31m} - (\delta + 360^o)\right| \leq \Delta\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 180^o)\right| \leq 2\Delta\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 540^o)\right| \leq 2\Delta\lambda$

No referencial $\lambda_{12}0\lambda_{23}$, para os cinco tipos de configurações de balizas, estas condições são verdadeiras para os pontos (λ_{12m}, λ_{31m}) que se encontram sobre as regiões sombreadas a verde (incluindo os contornos) na Figura 5.63, na Figura 5.64, na Figura 5.67, na Figura 5.69 e na Figura 5.71.

O método desenvolvido para calcular os erros máximos de posição e de orientação só se pode aplicar quando a posição calculada do robô se encontra no interior de alguma das regiões sombreadas a amarelo claro na Figura 5.65, na Figura 5.66, na Figura 5.68, na Figura 5.70 e na Figura 5.72.

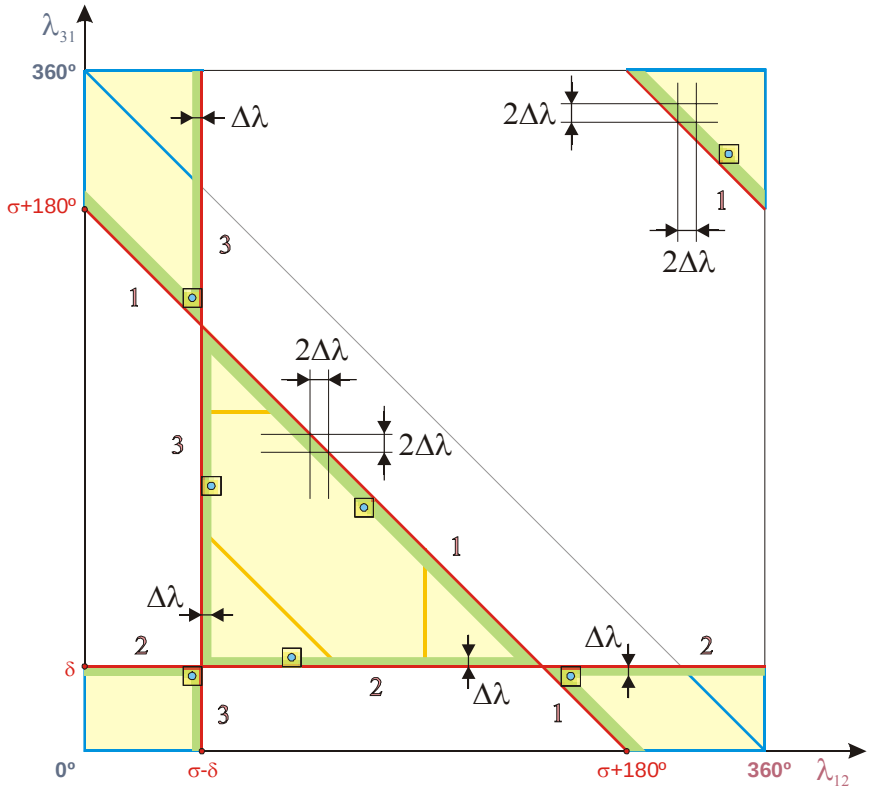


Figura 5.63: Com três balizas não colineares ordenadas no sentido directo, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

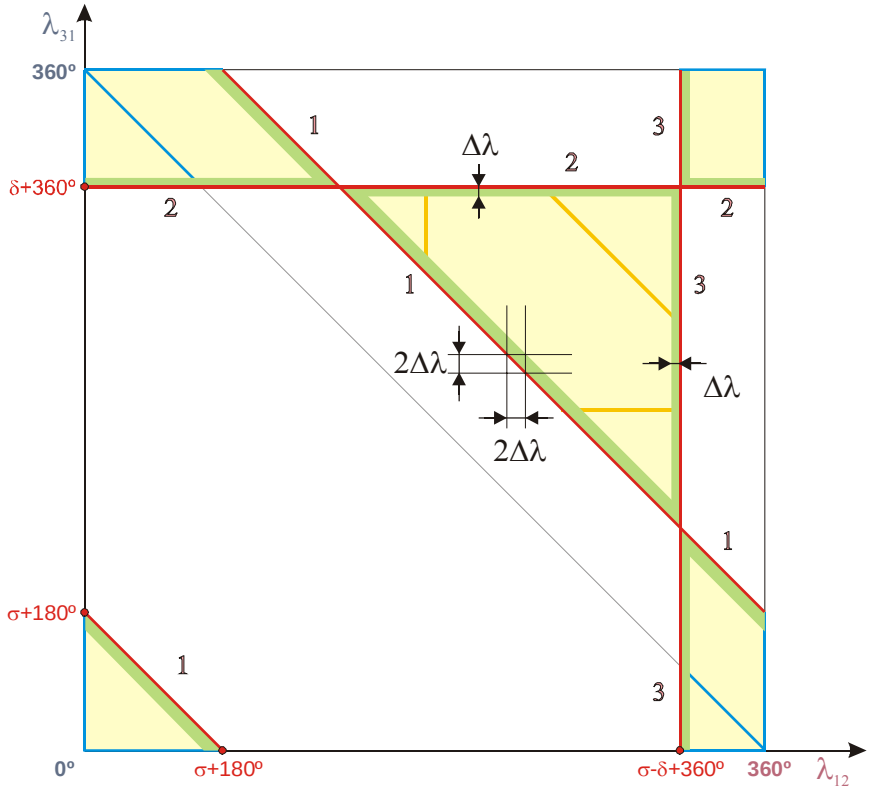


Figura 5.64: Com três balizas não colineares ordenadas no sentido inverso, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

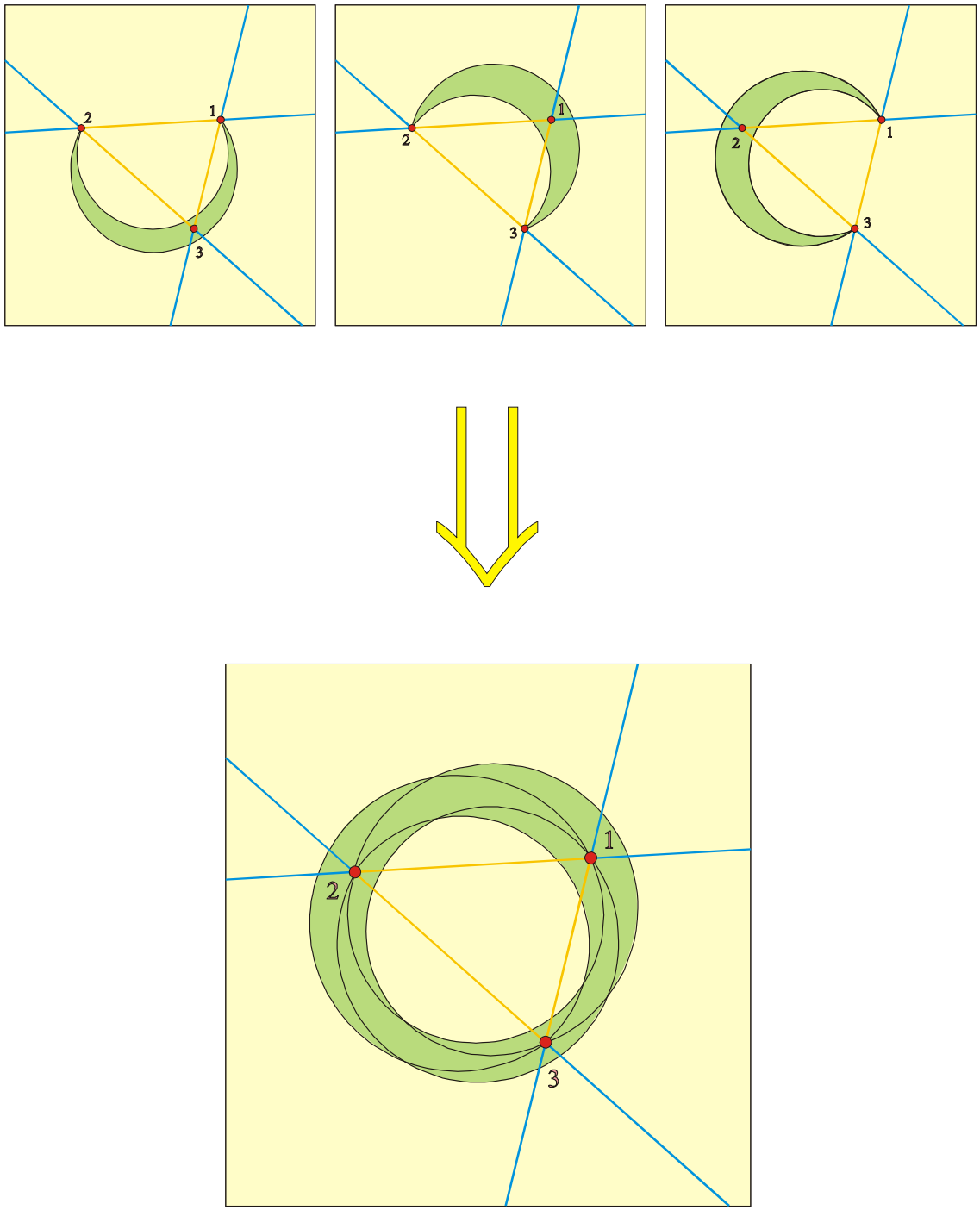


Figura 5.65: Com três balizas não colineares ordenadas no sentido directo, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

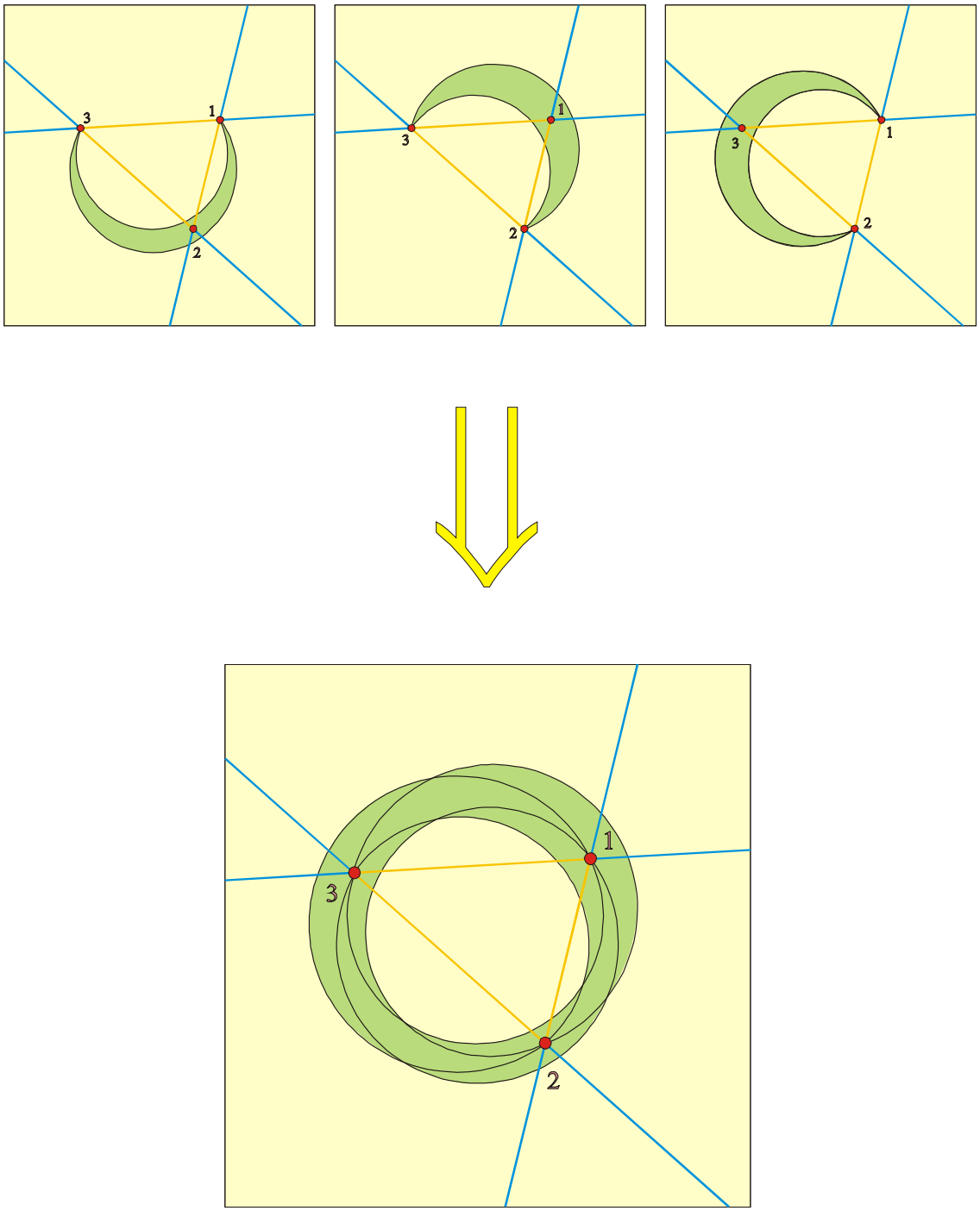


Figura 5.66: Com três balizas não colineares ordenadas no sentido inverso, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

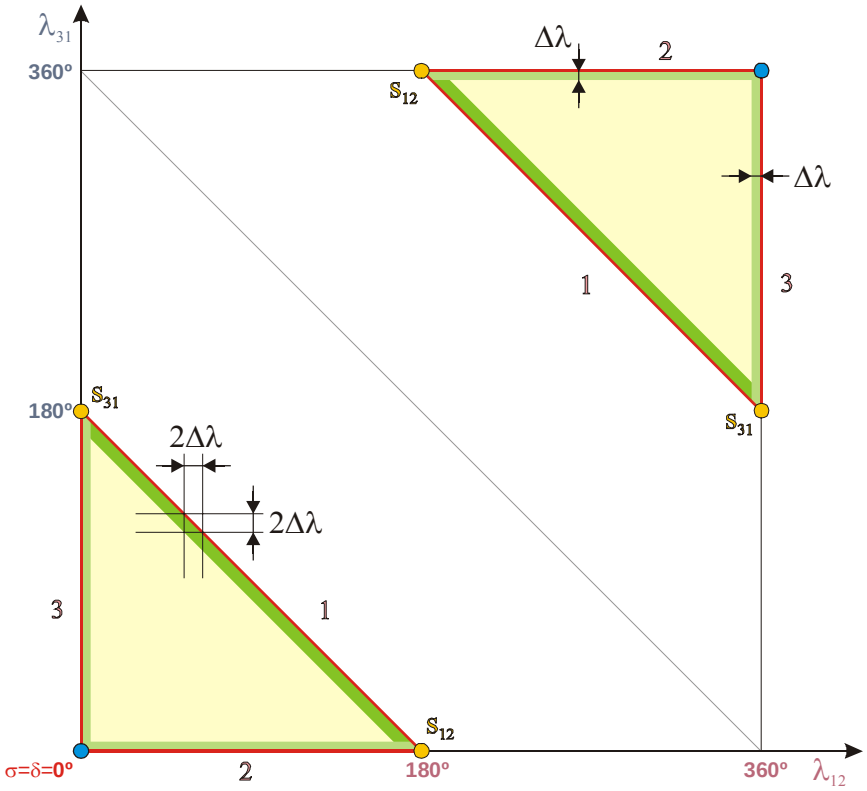


Figura 5.67: Com três balizas colineares e a baliza 1 entre as balizas 2 e 3, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

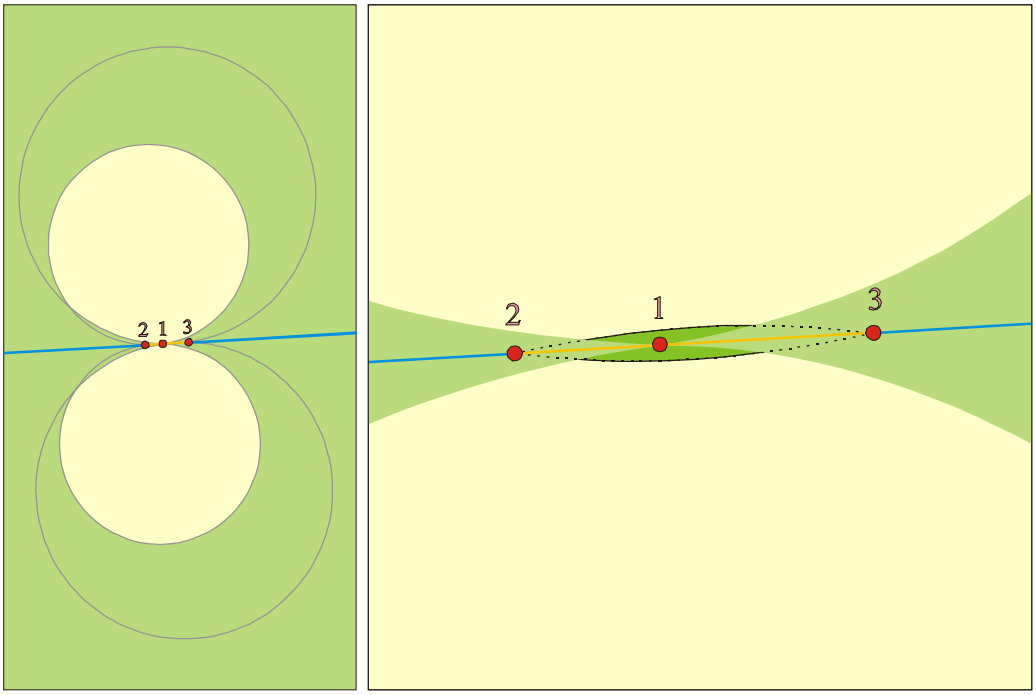


Figura 5.68: Com três balizas colineares e a baliza 1 entre as balizas 2 e 3, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

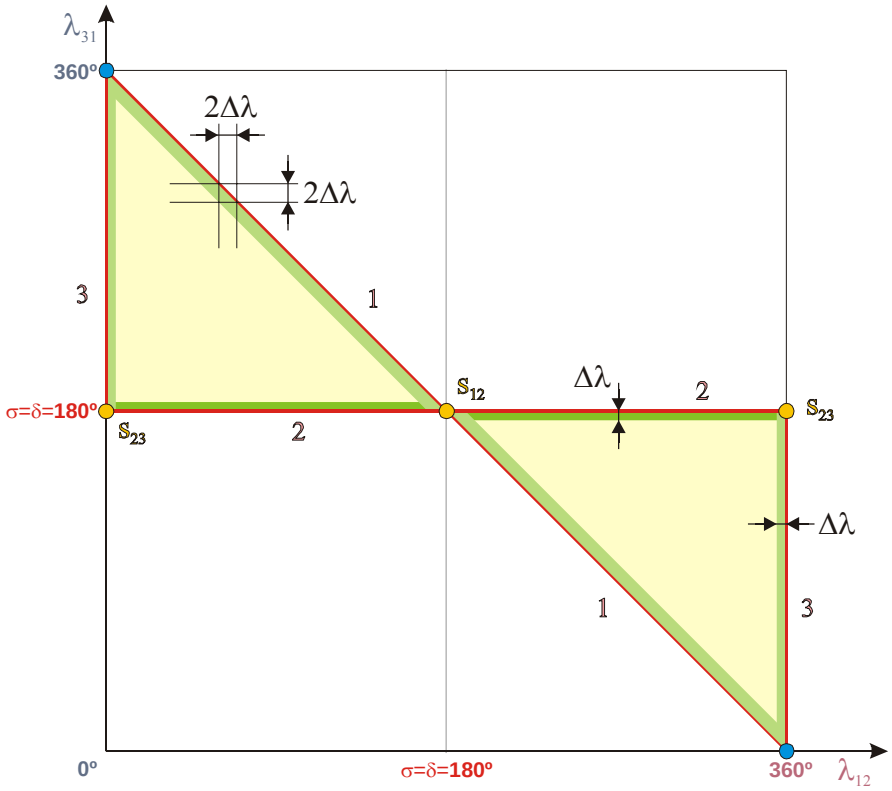


Figura 5.69: Com três balizas colineares e a baliza 2 entre as balizas 1 e 3, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

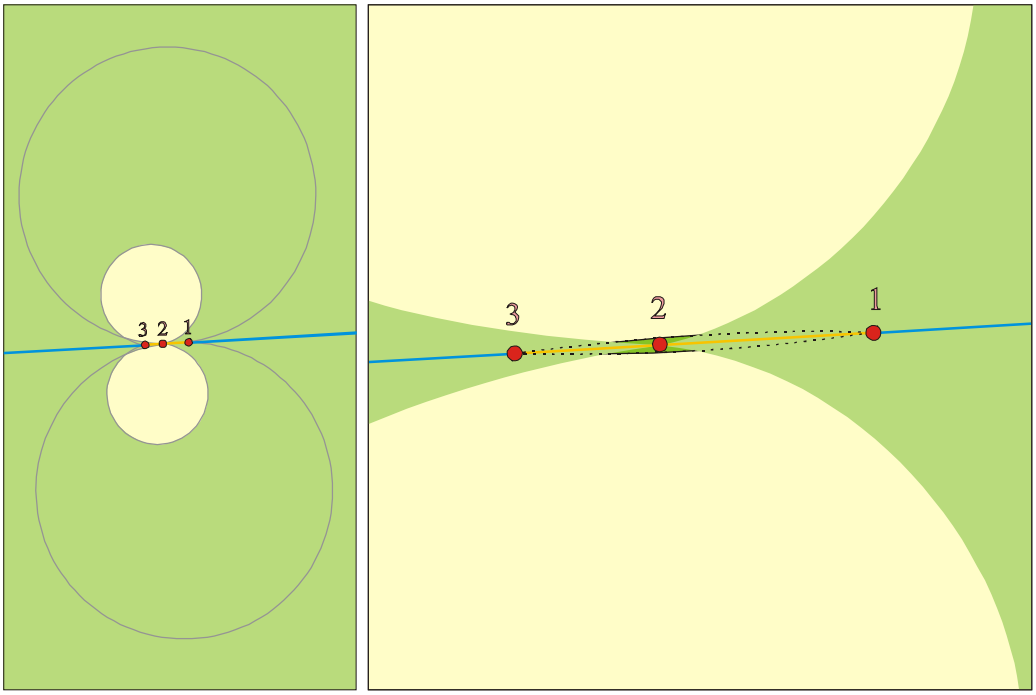


Figura 5.70: Com três balizas colineares e a baliza 2 entre as balizas 1 e 3, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

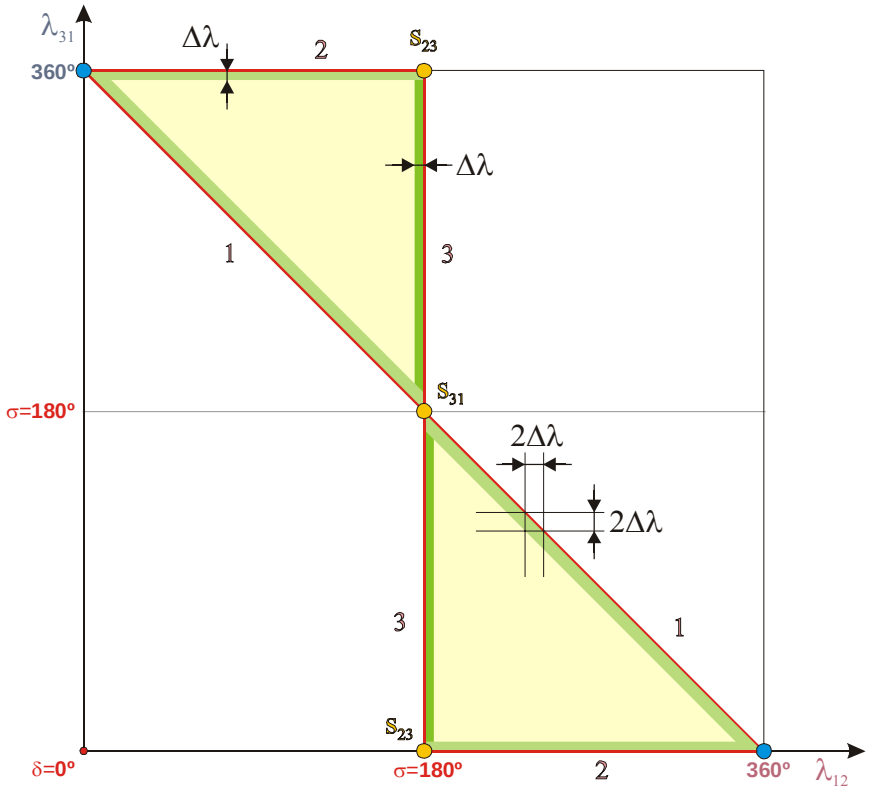


Figura 5.71: Com três balizas colineares e a baliza 3 entre as balizas 1 e 2, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

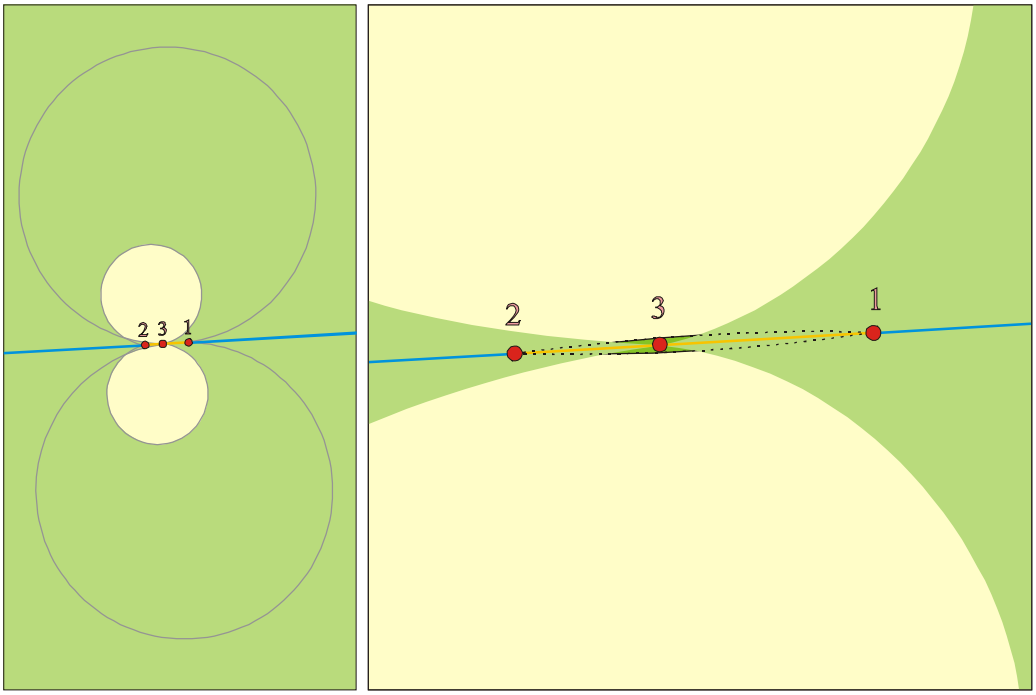


Figura 5.72: Com três balizas colineares e a baliza 3 entre as balizas 1 e 2, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} resultam de medições independentes de λ_{12} e λ_{31}, efectuadas com uma incerteza $\pm\Delta\lambda$.

Se λ_{12m} e λ_{31m} forem calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3 realizadas com uma incerteza $\pm\Delta\lambda$, para garantir que todos os vértices da superfície de incerteza de medição possuem imagens no plano de navegação e que nenhuma dessas imagens coincide com a posição de uma das três balizas utilizadas, é necessário e suficiente que não se verifique nenhuma das seguintes condições:

- Para três balizas não colineares:
  - $\left|\lambda_{12m}-(\sigma-\delta)\right| \le 2\Delta\lambda$
  - $\left|\lambda_{12m}-(\sigma-\delta+360^\circ)\right| \le 2\Delta\lambda$
  - $\left|\lambda_{31m}-\delta\right| \le 2\Delta\lambda$ (5.21)
  - $\left|\lambda_{31m}-(\delta+360^\circ)\right| \le 2\Delta\lambda$
  - $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+180^\circ)\right| \le 2\Delta\lambda$
  - $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+540^\circ)\right| \le 2\Delta\lambda$
- Para três balizas colineares e a baliza 1 entre as balizas 2 e 3 (σ=0º):
  - $\lambda_{12m} \le 2\Delta\lambda$ [35]
  - $\lambda_{12m} \ge 360^\circ-2\Delta\lambda$ [36]
  - $\lambda_{31m} \le 2\Delta\lambda$ [37] (5.22)
  - $\lambda_{31m} \ge 360^\circ-2\Delta\lambda$ [38]
  - $\left|(\lambda_{12m}+\lambda_{31m})-180^\circ\right| \le 2\Delta\lambda$ [39]
  - $\left|(\lambda_{12m}+\lambda_{31m})-540^\circ\right| \le 2\Delta\lambda$ [40]
- Para três balizas colineares e a baliza 2 entre as balizas 1 e 3 (σ=180º e δ=180º):
  - $\lambda_{12m} \le 2\Delta\lambda$ [35]
  - $\lambda_{12m} \ge 360^\circ-2\Delta\lambda$ [36] (5.23)
  - $\left|\lambda_{31m}-180^\circ\right| \le 2\Delta\lambda$ [37]
  - $\left|(\lambda_{12m}+\lambda_{31m})-360^\circ\right| \le 2\Delta\lambda$ [39]
- Para três balizas colineares e a baliza 3 entre as balizas 1 e 2 (σ=180º e δ=0º):
  - $\left|\lambda_{12m}-180^\circ\right| \le 2\Delta\lambda$ [35]
  - $\lambda_{31m} \le 2\Delta\lambda$ [37] (5.24)
  - $\lambda_{31m} \ge 360^\circ-2\Delta\lambda$ [38]
  - $\left|(\lambda_{12m}+\lambda_{31m})-360^\circ\right| \le 2\Delta\lambda$ [39]

[35] É um caso particular de $\left|\lambda_{12m}-(\sigma-\delta)\right| \le 2\Delta\lambda$.

[36] É um caso particular de $\left|\lambda_{12m}-(\sigma-\delta+360^\circ)\right| \le 2\Delta\lambda$.

[37] É um caso particular de $\left|\lambda_{31m}-\delta\right| \le 2\Delta\lambda$.

[38] É um caso particular de $\left|\lambda_{31m}-(\delta+360^\circ)\right| \le 2\Delta\lambda$.

[39] É um caso particular de $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+180^\circ)\right| \le 2\Delta\lambda$

[40] É um caso particular de $\left|(\lambda_{12m}+\lambda_{31m})-(\sigma+540^\circ)\right| \le 2\Delta\lambda$

Assim, com qualquer configuração de balizas, para ser possível efectuar a autolocalização e determinar os erros máximos de posição e de orientação quando a posição calculada do robô no plano de navegação é a imagem do ponto (λ_{12m}, λ_{31m}) no referencial $\lambda_{12}0\lambda_{23}$, supondo que λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3 efectuadas com uma incerteza $\pm\Delta\lambda$, é necessário e suficiente que não se verifique nenhuma das seguintes condições:

- $\left|\lambda_{12m} - (\sigma - \delta)\right| \leq 2\Delta\lambda$
- $\left|\lambda_{12m} - (\sigma - \delta + 360^o)\right| \leq 2\Delta\lambda$
- $\left|\lambda_{31m} - \delta\right| \leq 2\Delta\lambda$ (5.25)
- $\left|\lambda_{31m} - (\delta + 360^o)\right| \leq 2\Delta\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 180^o)\right| \leq 2\Delta\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 540^o)\right| \leq 2\Delta\lambda$

No referencial $\lambda_{12}0\lambda_{23}$, para os cinco tipos de configurações de balizas, estas condições são verdadeiras para os pontos (λ_{12m}, λ_{31m}) que se encontram sobre as regiões sombreadas a verde (incluindo os contornos) na Figura 5.73, na Figura 5.74, na Figura 5.77, na Figura 5.79 e na Figura 5.81.

O método desenvolvido para calcular os erros máximos de posição e de orientação só se pode aplicar quando a posição calculada do robô se encontra no interior de alguma das regiões sombreadas a amarelo claro na Figura 5.75, na Figura 5.76, na Figura 5.78, na Figura 5.80 e na Figura 5.82.

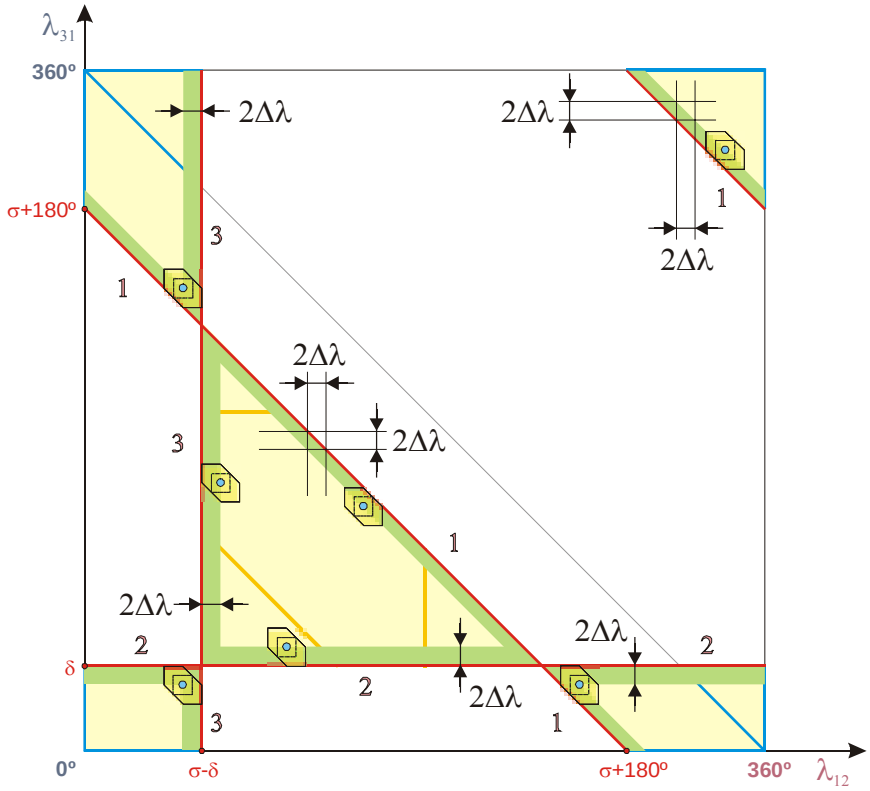


Figura 5.73: Com três balizas não colineares ordenadas no sentido directo, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

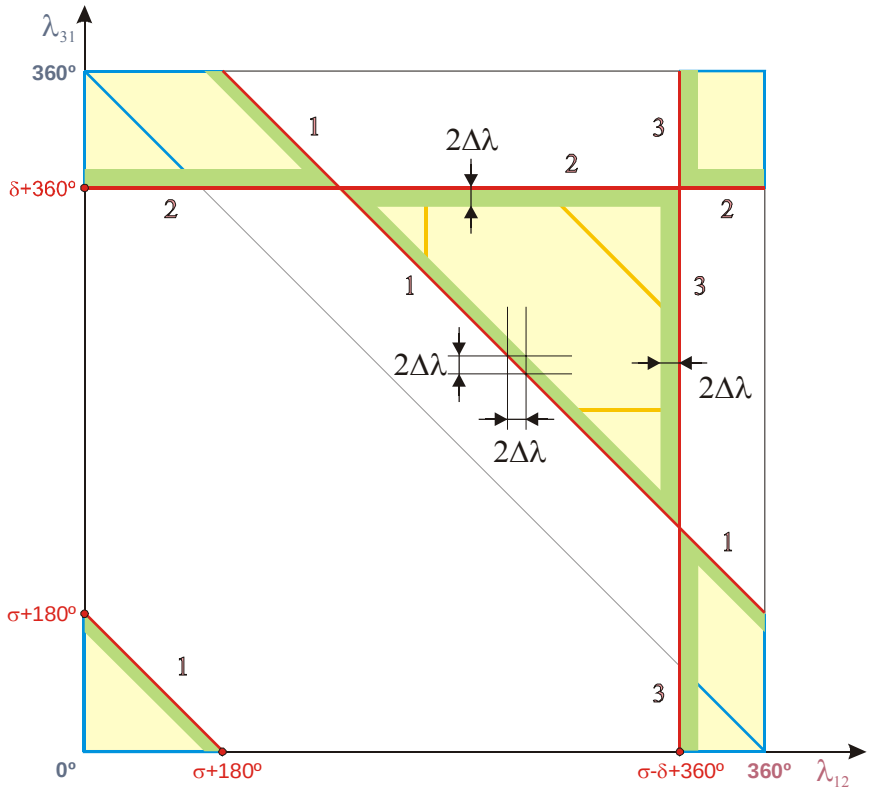


Figura 5.74: Com três balizas não colineares ordenadas no sentido inverso, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

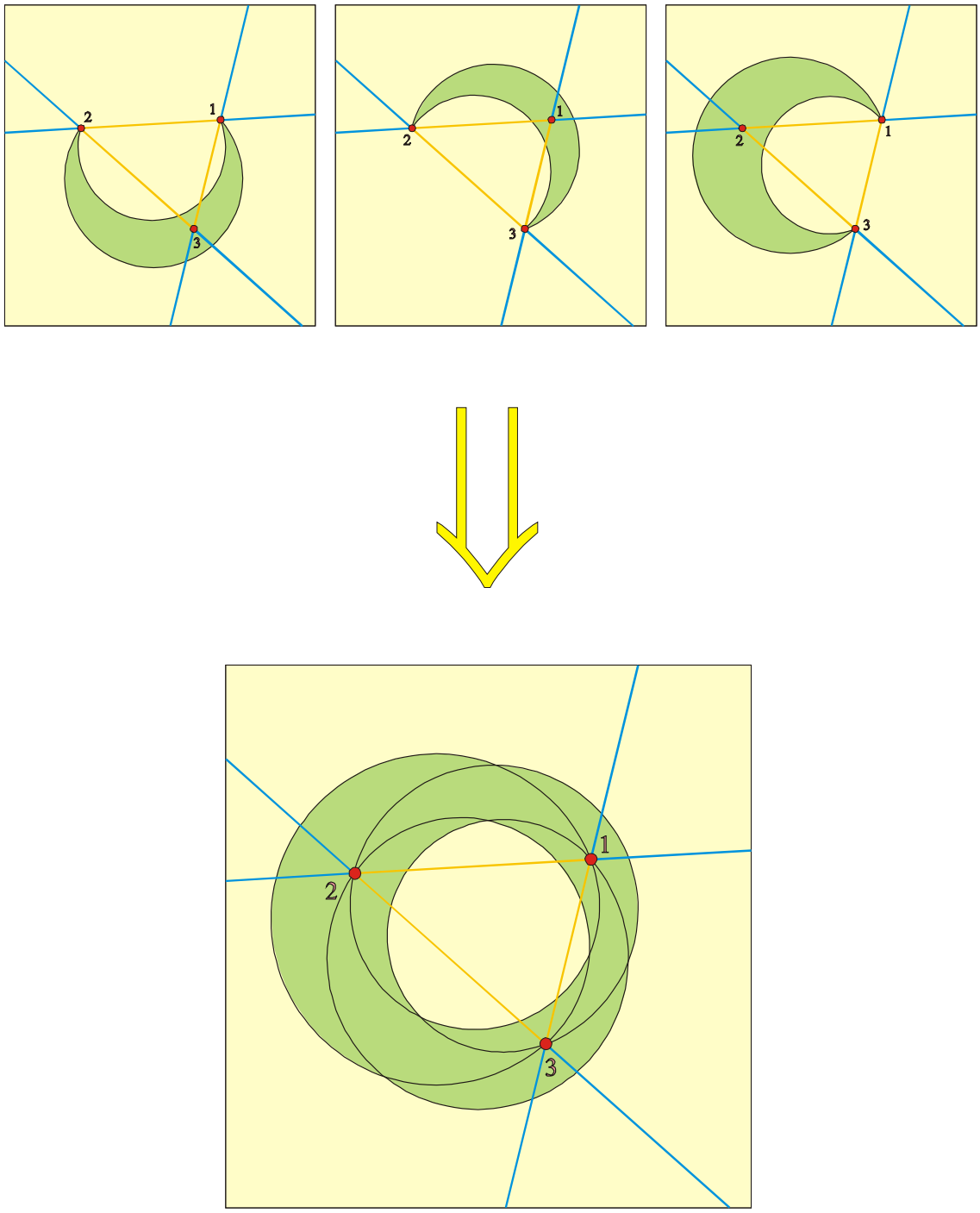


Figura 5.75: Com três balizas não colineares ordenadas no sentido directo, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

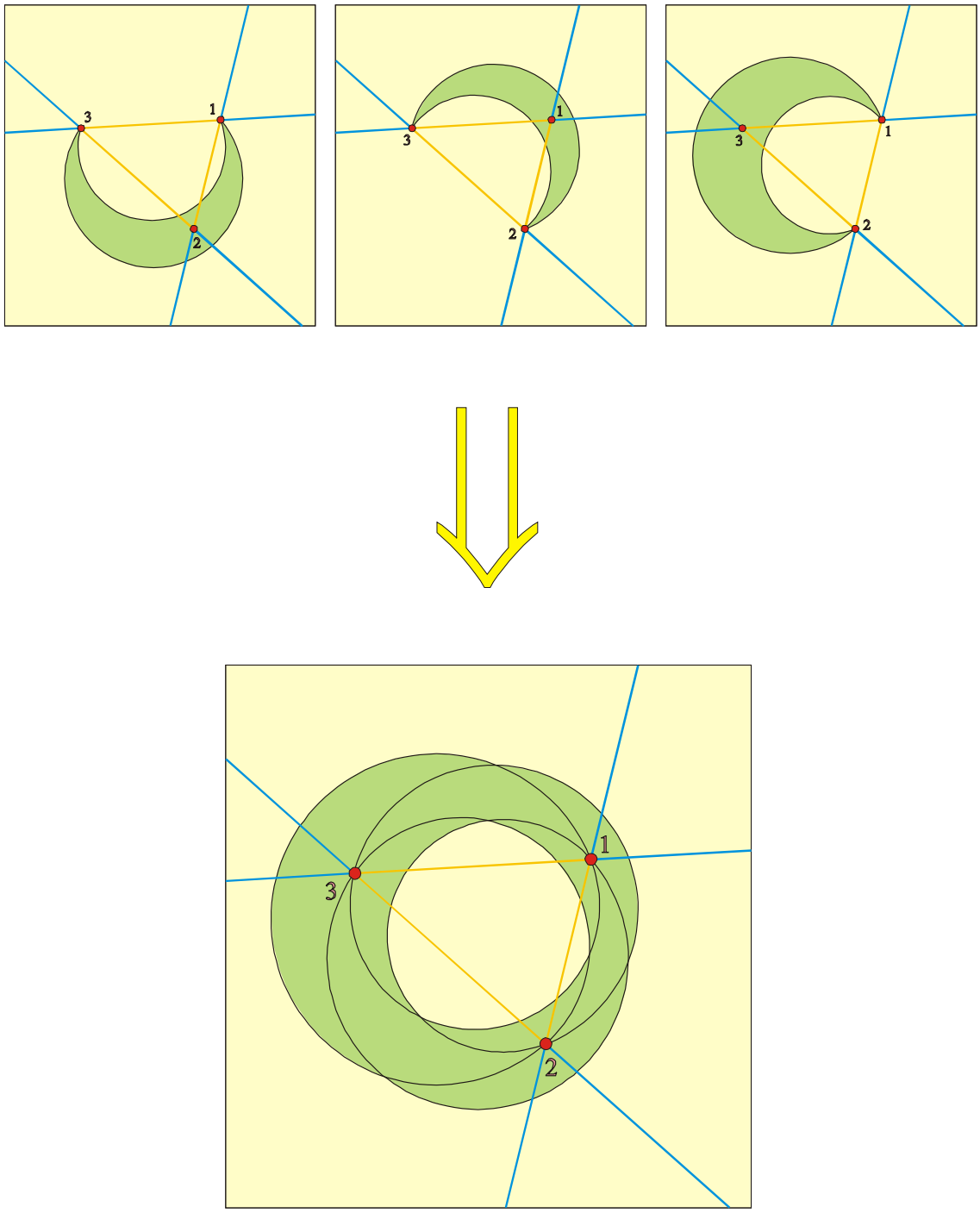


Figura 5.76: Com três balizas não colineares ordenadas no sentido inverso, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

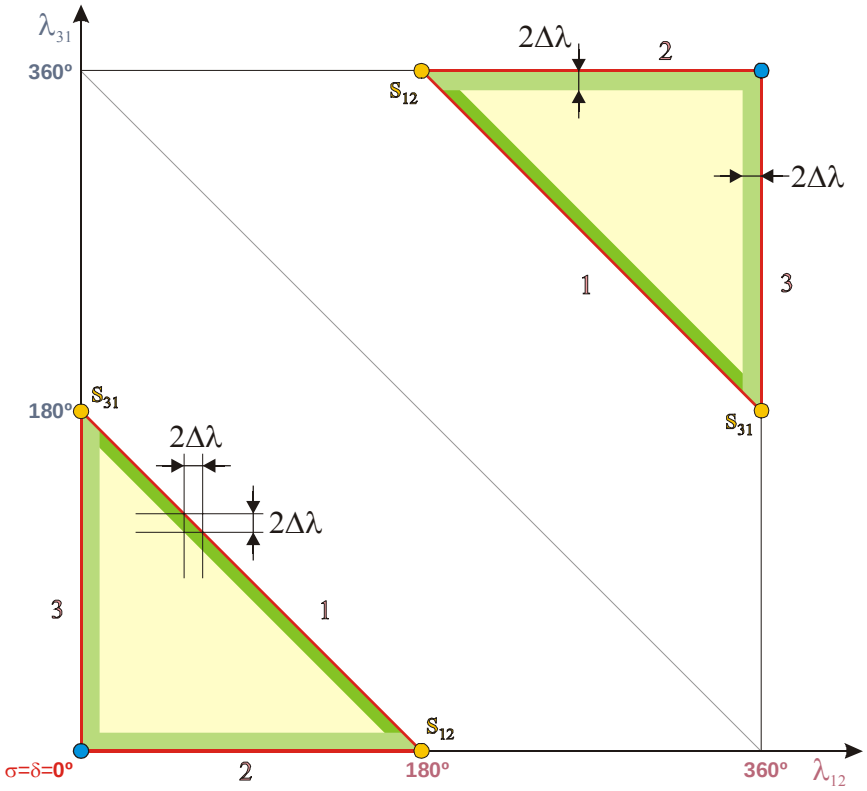


Figura 5.77: Com três balizas colineares e a baliza 1 entre as balizas 2 e 3, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

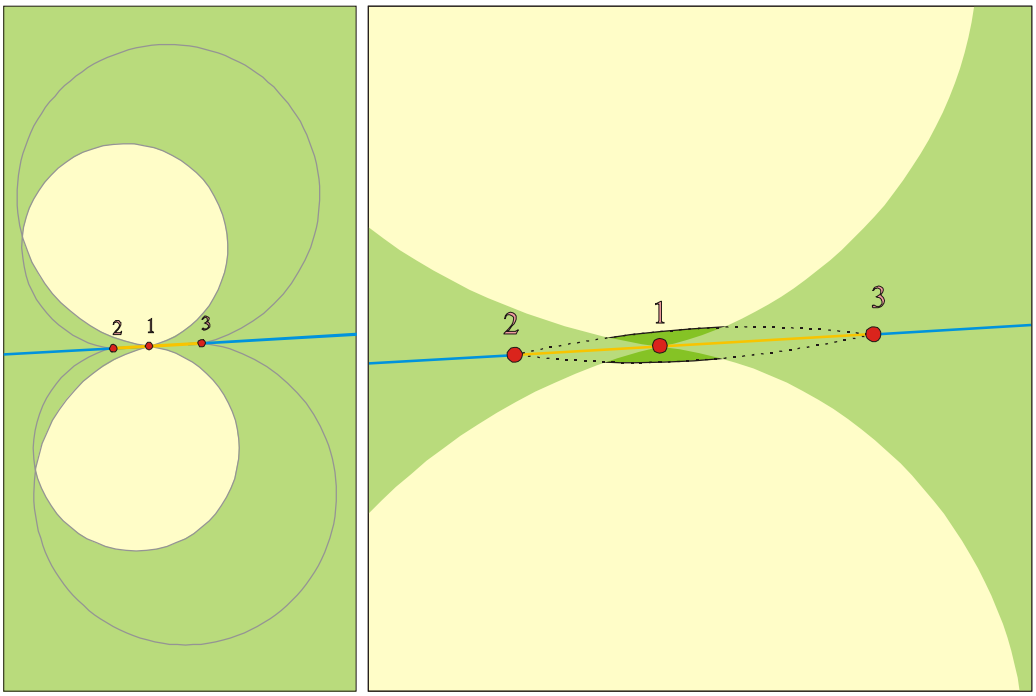


Figura 5.78: Com três balizas colineares e a baliza 1 entre as balizas 2 e 3, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

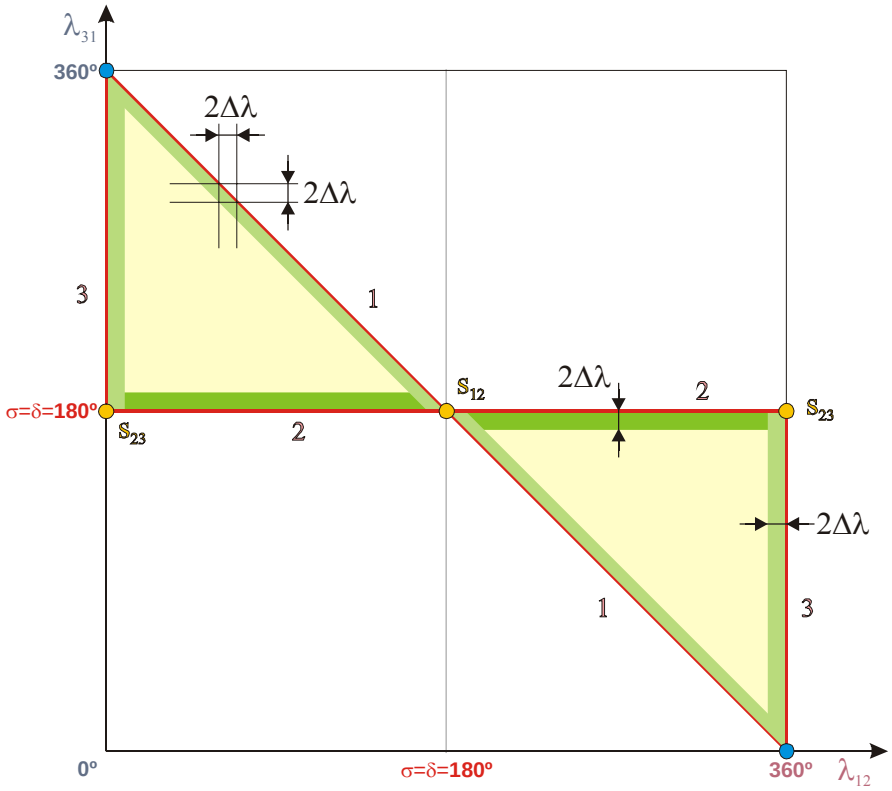


Figura 5.79: Com três balizas colineares e a baliza 2 entre as balizas 1 e 3, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

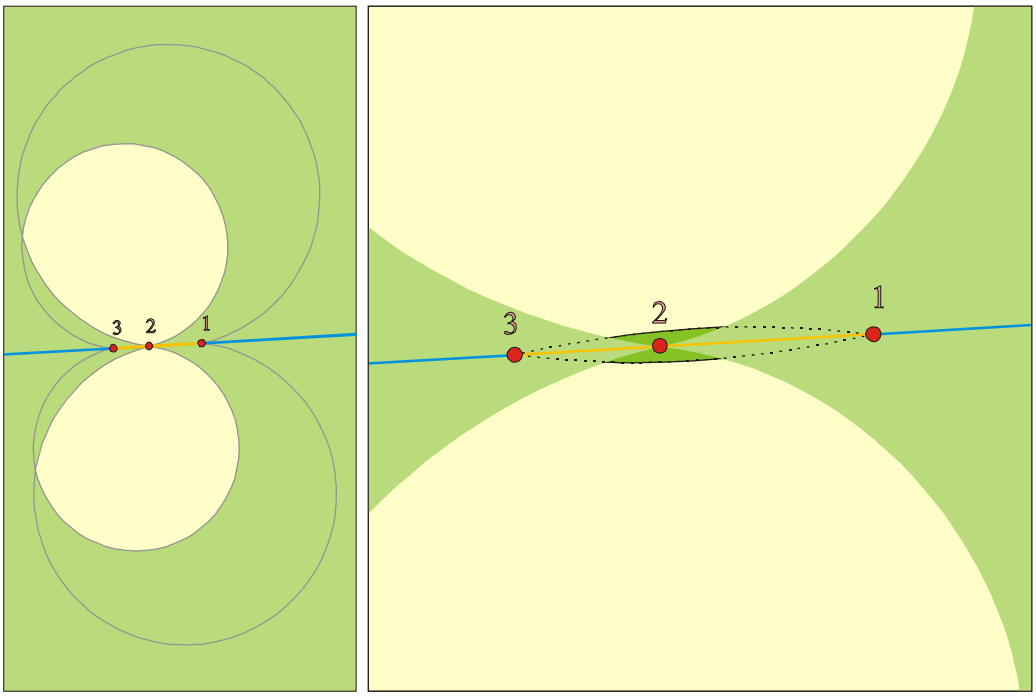


Figura 5.80: Com três balizas colineares e a baliza 2 entre as balizas 1 e 3, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

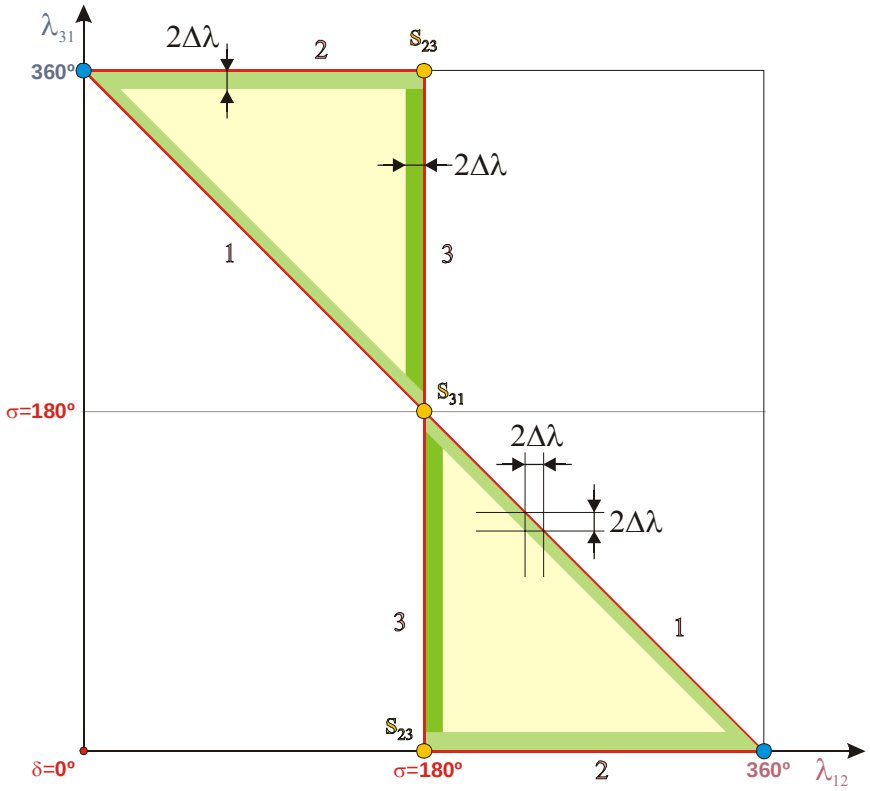


Figura 5.81: Com três balizas colineares e a baliza 3 entre as balizas 1 e 2, se o ponto (λ_{12m}, λ_{31m}) se encontrar sobre alguma das regiões sombreadas a verde ou seus contornos, não é possível calcular os erros máximos de posição e de orientação recorrendo ao método proposto. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

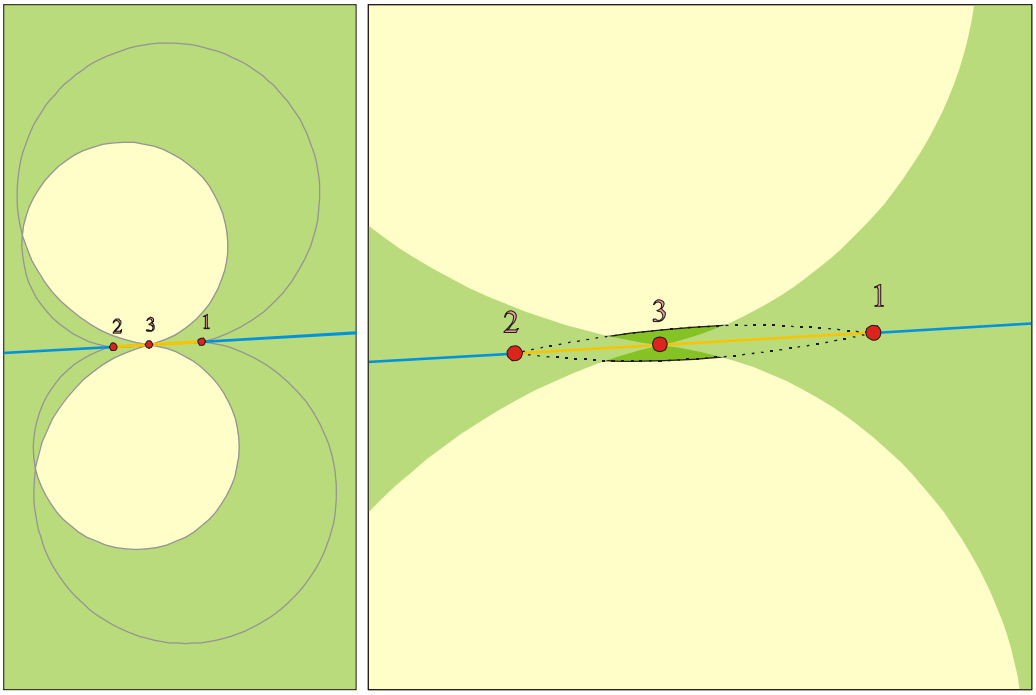


Figura 5.82: Com três balizas colineares e a baliza 3 entre as balizas 1 e 2, o método proposto para calcular os erros máximos de posição e de orientação só se pode aplicar se a posição calculada do robô se encontrar no interior de alguma das regiões sombreadas a amarelo claro. Os valores de λ_{12m} e λ_{31m} são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3, efectuadas com uma incerteza $\pm\Delta\lambda$.

5.6.5 Incerteza de Posição Devida aos Erros Aleatórios de Medição

A análise apresentada neste ponto refere-se apenas à incerteza de posição devida aos erros aleatórios de medição. Parte-se do princípio que não ocorrem erros grosseiros e que o instrumento de medição de ângulos foi devidamente calibrado, reduzindo os erros sistemáticos a valores insignificantes.

Se cada ângulo puder ser medido um número de vezes suficientemente elevado, torna-se possível efectuar o tratamento estatístico das medidas obtidas. λ_1, λ_2, λ_3, λ_{12} e λ_{31} passam a representar variáveis aleatórias cujos valores mais prováveis são afectados de incertezas que se supõe serem conhecidas.

Pode haver uma probabilidade de 100% de os erros aleatórios de medição não ultrapassarem certos limites, como acontece se λ_1, λ_2, λ_3, λ_{12} e λ_{31} forem variáveis aleatórias com distribuição uniforme de probabilidade. É esta a distribuição que resulta da resolução finita que caracteriza os instrumentos digitais de medição (Adams, 2002) – por exemplo, um goniómetro baseado num *encoder*. A determinação do erro máximo de posição em cada ponto do plano de navegação resume-se à aplicação directa do método anteriormente descrito.

Mas é muito frequente que λ_1, λ_{12} e λ_{31} sejam variáveis aleatórias com distribuição normal. Neste caso, não é possível estabelecer um limite máximo finito para os erros aleatórios de medição e a incerteza de posição só pode ser determinada com níveis de confiança inferiores a 100%. A incerteza devida aos erros aleatórios diminui à medida que o número de medições aumenta. Mas a cada medição também está sempre associada uma incerteza residual que se deve aos erros sistemáticos e que não é afectada pelo número de medições realizadas. Por isso, a partir de um certo ponto torna-se inútil aumentar o número de medições para reduzir a incerteza devida aos erros aleatórios. Além disso, pode não haver tempo para efectuar muitas medições. E, se o robô estiver em movimento, à medida que aumenta o número de medições de cada ângulo torna-se cada vez mais difícil de admitir como boa aproximação que todas as medições são efectuadas num mesmo ponto do plano de navegação. O tratamento estatístico também é incapaz de produzir informação útil sobre erros grandes isolados.

Seja $\lambda_m \pm \sigma_\lambda$ o resultado de várias medições de um ângulo λ, em que λ_m é o valor mais provável de λ e $\pm\sigma_\lambda$ é a incerteza associada a λ_m. Se o número de medições for

suficientemente elevado, o habitual é λ_m ser a média das medidas de λ e σ_λ o erro padrão (desvio padrão da média) dado por (Abreu *et al.*, 1994; Taylor, 1997):

$$\sigma_\lambda = \frac{\sigma}{\sqrt{n}} \tag{5.26}$$

em que σ é o desvio padrão[41] de cada medida de λ e n é o número de medições realizado.

Sejam λ_{12} e λ_{31} variáveis aleatórias e $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ duas funções de λ_{12} e λ_{31} utilizadas para calcular a posição do robô no referencial x0y definido no plano de navegação. As coordenadas x e y de cada ponto do plano são dadas por

$$\begin{aligned} x &= f(\lambda_{12}, \lambda_{31}) \\ y &= g(\lambda_{12}, \lambda_{31}) \end{aligned} \tag{5.27}$$

Sejam λ_{12m} e λ_{31m} os valores mais prováveis de λ_{12} e λ_{31}. Sejam x_R e y_R as coordenadas da posição verdadeira do robô P_R. Sejam x_{Rcalc} e y_{Rcalc}, coordenadas da posição mais provável do robô P_{Rcalc}, os valores mais prováveis de x_R e y_R, dados por

$$\begin{aligned} x_{Rcalc} &= f(\lambda_{12m}, \lambda_{31m}) \\ y_{Rcalc} &= g(\lambda_{12m}, \lambda_{31m}) \end{aligned} \tag{5.28}$$

A incerteza de posição devida aos erros aleatórios de medição é geralmente caracterizada pela *matriz de covariância de x e y,* $[\mathbf{C}_{xy}]$:

$$[\mathbf{C}_{xy}] = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix} \tag{5.29}$$

em que σ_x e σ_y são os *desvios padrão de x_{Rcalc} e y_{Rcalc},* ρ_{xy} é o *coeficiente de correlação entre x e y,* e σ_{xy} é a *covariância de x e y,* dada por

$$\sigma_{xy} = \rho_{xy} \cdot \sigma_x \cdot \sigma_y \tag{5.30}$$

O que se segue é o método habitualmente utilizado para determinar $[\mathbf{C}_{xy}]$ (Kaplan, 1996; Brown e Hwang, 1997; Taylor, 1997; Arras, 1998).

[41] Não confundir o desvio padrão σ com o ângulo σ utilizado para caracterizar a configuração de balizas. Esta página é a única em que σ se utiliza no primeiro contexto.

Sejam $\sigma_{\lambda 12}$ e $\sigma_{\lambda 31}$ os *desvios padrão de λ_{12m} e λ_{31m}*, ρ_λ o *coeficiente de correlação entre λ_{12} e λ_{31}*, e $\sigma_{\lambda 12 \lambda 31}$ a *covariância de λ_{12} e λ_{31}*, dada por

$$\sigma_{\lambda 12 \lambda 31} = \rho_\lambda \cdot \sigma_{\lambda 12} \cdot \sigma_{\lambda 31} \tag{5.31}$$

Seja $[\mathbf{C}_{\lambda 12 \lambda 31}]$ a *matriz de covariância de λ_{12} e λ_{31}:*

$$\left[\mathbf{C}_{\lambda 12 \lambda 31}\right] = \begin{bmatrix} \sigma_{\lambda 12}^2 & \sigma_{\lambda 12 \lambda 31} \\ \sigma_{\lambda 12 \lambda 31} & \sigma_{\lambda 31}^2 \end{bmatrix} \tag{5.32}$$

Seja $[\mathbf{J}]$ a matriz Jacobiana das funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$:

$$\left[\mathbf{J}\right] = \begin{bmatrix} \dfrac{\partial f(\lambda_{12},\lambda_{31})}{\partial \lambda_{12}} & \dfrac{\partial f(\lambda_{12},\lambda_{31})}{\partial \lambda_{31}} \\ \dfrac{\partial g(\lambda_{12},\lambda_{31})}{\partial \lambda_{12}} & \dfrac{\partial g(\lambda_{12},\lambda_{31})}{\partial \lambda_{31}} \end{bmatrix} \tag{5.33}$$

Se for válido aproximar as funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ pelas respectivas séries de Taylor de primeira ordem na vizinhança do ponto $(\lambda_{12m}, \lambda_{31m})$, então o processo habitual de obter $[\mathbf{C}_{xy}]$ é recorrer à expressão

$$\left[\mathbf{C}_{xy}\right] = \left[\mathbf{J}_m\right] \cdot \left[\mathbf{C}_{\lambda 12 \lambda 31}\right] \cdot \left[\mathbf{J}_m\right]^T \tag{5.34}$$

em que $[\mathbf{J}_m]$ é a matriz $[\mathbf{J}]$ no ponto $(\lambda_{12m}, \lambda_{31m})$ do referencial $\lambda_{12}0\lambda_{31}$ e $[\mathbf{J}_m]^T$ é a *matriz transposta* de $[\mathbf{J}_m]$.

Se λ_{12} e λ_{31} forem variáveis aleatórias com distribuição normal, então x e y também o são, e a sua densidade de probabilidade conjunta é dada por:

$$\begin{aligned} f_{xy}(x,y) &= \frac{1}{2\pi\sqrt{C_{xy}}} e^{-\frac{1}{2}\left(\left[x - x_{Rcalc} \quad y - y_{Rcalc} \right] \left[\mathbf{C}_{xy}\right]^{-1} \cdot \begin{bmatrix} x - x_{Rcalc} \\ y - y_{Rcalc} \end{bmatrix} \right)} \\ &= \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho_{xy}^2}} e^{-\frac{1}{2(1-\rho_{xy}^2)}\left[\frac{(x-x_{Rcalc})^2}{\sigma_x^2} - \frac{2\rho_{xy}(x-x_{Rcalc})(y-y_{Rcalc})}{\sigma_x\sigma_y} + \frac{(y-y_{Rcalc})^2}{\sigma_y^2} \right]} \\ &= \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho_{xy}^2}} e^{-\frac{q^2}{2}} \end{aligned} \tag{5.35}$$

em que $[\mathbf{C}_{xy}]^{-1}$ e C_{xy} são, respectivamente, a matriz inversa e o determinante da matriz $[\mathbf{C}_{xy}]$ e q é um número tal que

$$q^2 = \frac{1}{1-\rho_{xy}^2}\left[\frac{(x-x_{Rcalc})^2}{\sigma_x^2} - \frac{2\rho_{xy}(x-x_{Rcalc})(y-y_{Rcalc})}{\sigma_x\sigma_y} + \frac{(y-y_{Rcalc})^2}{\sigma_y^2}\right] \tag{5.36}$$

Se q for constante, a expressão anterior é a equação de uma elipse situada no referencial x0y, centrada em P_{Rcalc} (x_{Rcalc}, y_{Rcalc}) e cujas dimensões aumentam com o aumento de q. A probabilidade de P_R (x_R, y_R) se situar dentro dessa elipse é dada por

$$p_{xy} = 1 - e^{-\frac{q^2}{2}} \tag{5.37}$$

Se q=1 então p_{xy}=39%. Se este nível de confiança for insuficiente, torna-se necessário aumentar q, o que implica o aumento das dimensões da elipse. Para obter um nível de confiança aproximadamente igual a 95% torna-se necessário fazer q^2=6.

Este método possui as seguintes limitações:

- Os valores de σ_x, σ_y e σ_{xy} que aparecem na matriz de covariância de x e y, são aproximados, uma vez que as funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ são aproximadas pelas respectivas séries de Taylor de primeira ordem na vizinhança do ponto $(\lambda_{12m}, \lambda_{31m})$.

- As expressões analíticas das derivadas parciais das funções $f(\lambda_{12}, \lambda_{31})$ e $g(\lambda_{12}, \lambda_{31})$ em ordem a λ_{12} e a λ_{31}, que aparecem em [**J**], podem ser extremamente difíceis de obter, tornando-se necessário efectuar mais aproximações num método que já é aproximado por natureza.

- Devido às aproximações inerentes ao método, este só é válido se $\sigma_{\lambda 12}$, $\sigma_{\lambda 31}$ e $\sigma_{\lambda 12\lambda 31}$ forem suficientemente pequenos.

Seguidamente apresenta-se o modo de adaptar o algoritmo apresentado no ponto 5.6.2 à caracterização da incerteza de posição devida aos erros aleatórios de medição.

Se λ_{12} e λ_{31} forem variáveis aleatórias com distribuição normal, a sua densidade de probabilidade conjunta é dada por (Brown e Hwang, 1997)

$$
\begin{aligned}
f_{\lambda 12\lambda 31}(\lambda_{12},\lambda_{31}) &= \frac{1}{2\pi\sqrt{C_{\lambda 12\lambda 31}}}\, e^{-\frac{1}{2}\left(\left[\lambda_{12}-\lambda_{12m} \quad \lambda_{31}-\lambda_{31m}\right]\left[\mathbf{C}_{\lambda 12\lambda 31}\right]^{-1}\begin{bmatrix}\lambda_{12}-\lambda_{12m}\\ \lambda_{31}-\lambda_{31m}\end{bmatrix}\right)} \\
&= \frac{1}{2\pi\sigma_{\lambda 12}\sigma_{\lambda 31}\sqrt{1-\rho^2_{\lambda 12\lambda 31}}}\, e^{-\frac{1}{2\left(1-\rho^2_{\lambda 12\lambda 31}\right)}\left[\frac{(\lambda_{12}-\lambda_{12m})^2}{\sigma^2_{\lambda 12}}-\frac{2\rho_{\lambda 12\lambda 31}(\lambda_{12}-\lambda_{12m})(\lambda_{31}-\lambda_{31m})}{\sigma_{\lambda 12}\sigma_{\lambda 31}}+\frac{(\lambda_{31}-\lambda_{31m})^2}{\sigma^2_{\lambda 31}}\right]} \\
&= \frac{1}{2\pi\sigma_{\lambda 12}\sigma_{\lambda 31}\sqrt{1-\rho^2_{\lambda 12\lambda 31}}}\, e^{-\frac{m^2}{2}}
\end{aligned}
\tag{5.38}
$$

em que $[\mathbf{C}_{\lambda 12\lambda 31}]^{-1}$ e $C_{\lambda 12\lambda 31}$ são, respectivamente, a matriz inversa e o determinante da matriz $[\mathbf{C}_{\lambda 12\lambda 31}]$ e m é um número tal que

$$
m^2 = \frac{1}{1-\rho^2_{\lambda 12\lambda 31}}\left[\frac{(\lambda_{12}-\lambda_{12m})^2}{\sigma^2_{\lambda 12}}-\frac{2\rho_{\lambda 12\lambda 31}(\lambda_{12}-\lambda_{12m})(\lambda_{31}-\lambda_{31m})}{\sigma_{\lambda 12}\sigma_{\lambda 31}}+\frac{(\lambda_{31}-\lambda_{31m})^2}{\sigma^2_{\lambda 31}}\right] \tag{5.39}
$$

Se m for constante, a expressão anterior é a equação de uma elipse situada no referencial $\lambda_{12}0\lambda_{31}$ e centrada no ponto de coordenadas λ_{12m} e λ_{31m} (imagem inversa do ponto P_{Rcalc}). A probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro dessa elipse é dada por (Kaplan, 1996)

$$
p_\lambda = 1 - e^{-\frac{m^2}{2}} \tag{5.40}
$$

Se λ_{12m} e λ_{31m} possuírem o mesmo desvio padrão σ_λ, ou seja, se o resultado das medições de λ_{12} e λ_{31} for

$$
\begin{cases}
\lambda_{12m} \pm \sigma_\lambda \\
\lambda_{31m} \pm \sigma_\lambda
\end{cases} \tag{5.41}
$$

isso significa que

$$
\begin{cases}
\sigma_{\lambda 12} = \sigma_\lambda \\
\sigma_{\lambda 31} = \sigma_\lambda
\end{cases} \tag{5.42}
$$

e se, além disso, λ_{12} e λ_{31} forem estatisticamente independentes, então

$$
\begin{cases}
\sigma_{\lambda 12\lambda 31} = 0 \\
\rho_\lambda = \dfrac{\sigma_{\lambda 12\lambda 31}}{\sigma_{\lambda 12}\cdot\sigma_{\lambda 31}} = 0
\end{cases} \tag{5.43}
$$

daí resultando que

$$[\mathbf{C}_{\lambda 12\lambda 31}] = \begin{bmatrix} \sigma_\lambda^2 & 0 \\ 0 & \sigma_\lambda^2 \end{bmatrix} \tag{5.44}$$

Neste caso particular, a densidade de probabilidade conjunta de λ_{12} e λ_{31} é dada por

$$f_{\lambda 12\lambda 31}(\lambda_{12}, \lambda_{31}) = \frac{1}{2\pi\sigma_\lambda^2} e^{-\frac{m^2}{2}} \tag{5.45}$$

em que

$$m^2 = \frac{1}{\sigma_\lambda^2}\left[(\lambda_{12} - \lambda_{12m})^2 + (\lambda_{31} - \lambda_{31m})^2\right] \tag{5.46}$$

ou seja,

$$m^2\sigma_\lambda^2 = (\lambda_{12} - \lambda_{12m})^2 + (\lambda_{31} - \lambda_{31m})^2 \tag{5.47}$$

Se m for constante, a expressão anterior é a equação de uma circunferência Φ (Figura 5.83) de raio $m\sigma_\lambda$ situada no referencial $\lambda_{12}0\lambda_{31}$ e centrada no ponto de coordenadas λ_{12m} e λ_{31m} (imagem inversa do ponto P_{Rcalc}). Esta circunferência encontra-se inscrita numa superfície de incerteza de medição que é um quadrado cujos vértices são os seguintes pontos:

$$\begin{aligned} &\mathbf{P1}\ (\lambda_{12m} - m\sigma_\lambda\ ,\lambda_{31m} - m\sigma_\lambda) \\ &\mathbf{P2}\ (\lambda_{12m} - m\sigma_\lambda\ ,\lambda_{31m} + m\sigma_\lambda) \\ &\mathbf{P3}\ (\lambda_{12m} + m\sigma_\lambda\ ,\lambda_{31m} + m\sigma_\lambda) \\ &\mathbf{P4}\ (\lambda_{12m} + m\sigma_\lambda\ ,\lambda_{31m} - m\sigma_\lambda) \end{aligned} \tag{5.48}$$

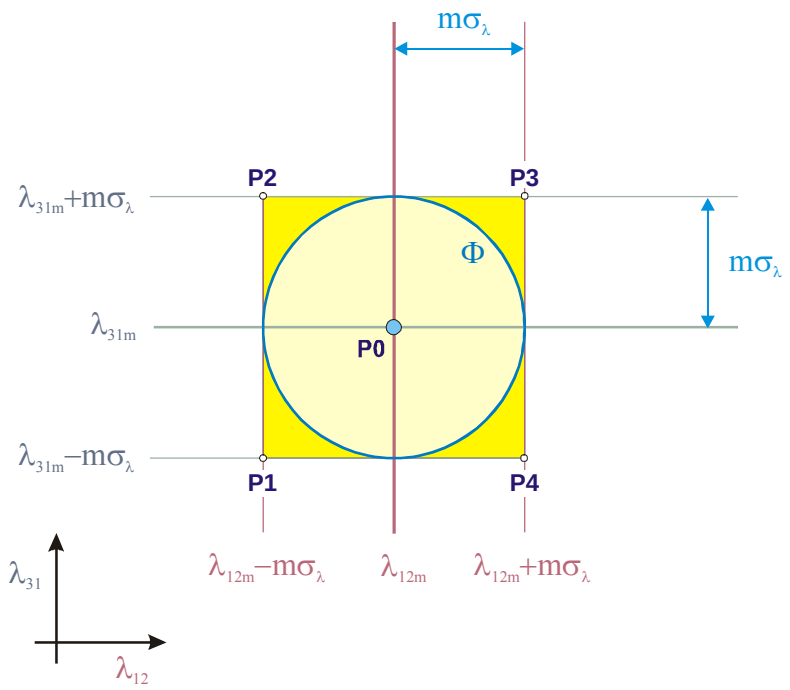


Figura 5.83: A probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são P1, P2, P3 e P4 é superior a p_λ, que é a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da circunferência de raio $m\sigma_\lambda$ centrada em P0.

Assim, a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro desta superfície de incerteza de medição é superior a p_λ e, por isso, a respectiva superfície de incerteza de posição possui uma probabilidade superior a p_λ de conter a posição verdadeira do robô, P_R.

A aplicação do algoritmo apresentado no ponto 5.6.2 para cálculo do erro máximo de posição a esta superfície de incerteza de posição produz um valor $\Delta P_{Rmáx}$ que, neste caso, tem o seguinte significado: *existe uma probabilidade superior a p_λ de que a distância entre a posição mais provável do robô P_{Rcalc} e a sua posição verdadeira P_R não ultrapasse o valor $\Delta P_{Rmáx}$.*

Para m=1, a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da circunferência Φ_{39}, representada na Figura 5.84, é de 39%:

$$m=1 \Rightarrow \begin{cases} p_\lambda = 1 - e^{-\frac{1}{2}} = 0{,}39 \\ \sigma_\lambda^2 = (\lambda_{12} - \lambda_{12m})^2 + (\lambda_{31} - \lambda_{31m})^2 \end{cases} \quad (5.49)$$

Assim, é superior a 39% a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são os seguintes pontos:

$$\begin{aligned}
&\mathbf{P1_{39}}\ \left(\lambda_{12m}-\sigma_\lambda\,,\lambda_{31m}-\sigma_\lambda\right)\\
&\mathbf{P2_{39}}\ \left(\lambda_{12m}-\sigma_\lambda\,,\lambda_{31m}+\sigma_\lambda\right)\\
&\mathbf{P3_{39}}\ \left(\lambda_{12m}+\sigma_\lambda\,,\lambda_{31m}+\sigma_\lambda\right)\\
&\mathbf{P4_{39}}\ \left(\lambda_{12m}+\sigma_\lambda\,,\lambda_{31m}-\sigma_\lambda\right)
\end{aligned} \tag{5.50}$$

Uma vez calculado $\Delta P_{Rmáx}$ pode afirmar-se que existe uma probabilidade superior a 39% de que a distância entre a posição mais provável do robô P_{Rcalc} e a sua posição verdadeira P_R não ultrapasse o valor $\Delta P_{Rmáx}$.

A probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da circunferência Φ_{95} (Figura 5.84) é de 95%, valor que se obtém para m^2=6:

$$p_\lambda = 0{,}95 \ \Rightarrow\ \begin{cases} m^2 = -2\ln 0{,}05 \approx 6 \\ 6\sigma_\lambda^2 = (\lambda_{12}-\lambda_{12m})^2 + (\lambda_{31}-\lambda_{31m})^2 \end{cases} \tag{5.51}$$

Neste caso, é superior a 95% probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são os seguintes pontos:

$$\begin{aligned}
&\mathbf{P1_{95}}\ \left(\lambda_{12m}-\sqrt{6}\sigma_\lambda\,,\lambda_{31m}-\sqrt{6}\sigma_\lambda\right)\\
&\mathbf{P2_{95}}\ \left(\lambda_{12m}-\sqrt{6}\sigma_\lambda\,,\lambda_{31m}+\sqrt{6}\sigma_\lambda\right)\\
&\mathbf{P3_{95}}\ \left(\lambda_{12m}+\sqrt{6}\sigma_\lambda\,,\lambda_{31m}+\sqrt{6}\sigma_\lambda\right)\\
&\mathbf{P4_{95}}\ \left(\lambda_{12m}+\sqrt{6}\sigma_\lambda\,,\lambda_{31m}-\sqrt{6}\sigma_\lambda\right)
\end{aligned} \tag{5.52}$$

Então, uma vez calculado $\Delta P_{Rmáx}$ pode afirmar-se que existe uma probabilidade superior a 95% de que a distância entre a posição mais provável do robô P_{Rcalc} e a sua posição verdadeira P_R não ultrapasse o valor $\Delta P_{Rmáx}$.

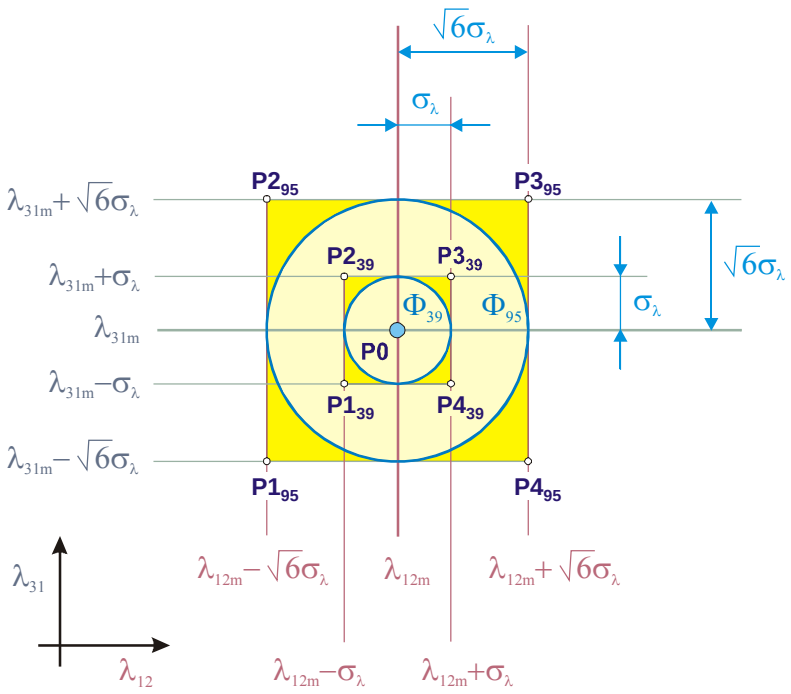


Figura 5.84: A probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são $P1_{39}$, $P2_{39}$, $P3_{39}$ e $P4_{39}$ é superior a 39%, que é a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da circunferência Φ_{39}. E a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são $P1_{95}$, $P2_{95}$, $P3_{95}$ e $P4_{95}$ é superior a 95%, que é a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da circunferência Φ_{95}.

Sejam λ_1, λ_2 e λ_3 variáveis aleatórias estatisticamente independentes e λ_{1m}, λ_{2m} e λ_{3m} os valores mais prováveis dessas variáveis. Se λ_{1m}, λ_{2m} e λ_{3m} possuírem o mesmo desvio padrão σ_λ, então o resultado das medições de λ_1, λ_2 e λ_3 é

$$\begin{cases} \lambda_{1m} \pm \sigma_\lambda \\ \lambda_{2m} \pm \sigma_\lambda \\ \lambda_{3m} \pm \sigma_\lambda \end{cases} \tag{5.53}$$

A matriz de covariância de λ_1, λ_2 e λ_3 é a seguinte:

$$[\mathbf{C}_{\lambda 1 \lambda 2 \lambda 3}] = \begin{bmatrix} \sigma_\lambda^2 & 0 & 0 \\ 0 & \sigma_\lambda^2 & 0 \\ 0 & 0 & \sigma_\lambda^2 \end{bmatrix} \tag{5.54}$$

Se λ_{12} e λ_{31} forem calculados a partir de λ_1, λ_2 e λ_3 recorrendo às expressões

$$\begin{cases} \lambda_{12} = \lambda_2 - \lambda_1 \quad \vee \quad \lambda_{12} = \lambda_2 - \lambda_1 + 360^\circ \\ \lambda_{31} = \lambda_1 - \lambda_3 \quad \vee \quad \lambda_{31} = \lambda_1 - \lambda_3 + 360^\circ \end{cases} \tag{5.55}$$

então a matriz Jacobiana de λ_{12} e λ_{31}, $[\mathbf{J}_\lambda]$, e a respectiva matriz transposta, $[\mathbf{J}_\lambda]^T$, são as seguintes:

$$[\mathbf{J}_\lambda]=\begin{bmatrix}\frac{\partial\lambda_{12}}{\partial\lambda_1} & \frac{\partial\lambda_{12}}{\partial\lambda_2} & \frac{\partial\lambda_{12}}{\partial\lambda_3}\\ \frac{\partial\lambda_{31}}{\partial\lambda_1} & \frac{\partial\lambda_{31}}{\partial\lambda_2} & \frac{\partial\lambda_{31}}{\partial\lambda_3}\end{bmatrix}=\begin{bmatrix}-1 & 1 & 0\\ 1 & 0 & -1\end{bmatrix} \tag{5.56}$$

$$[\mathbf{J}_\lambda]^T=\begin{bmatrix}-1 & 1\\ 1 & 0\\ 0 & -1\end{bmatrix} \tag{5.57}$$

Seguidamente, determina-se[42] a matriz de covariância de λ_{12} e λ_{31}:

$$[\mathbf{C}_{\lambda12\lambda31}]=[\mathbf{J}_\lambda]\cdot[\mathbf{C}_{\lambda1\lambda2\lambda2}]\cdot[\mathbf{J}_\lambda]^T=\begin{bmatrix}2\sigma_\lambda^2 & -\sigma_\lambda^2\\ -\sigma_\lambda^2 & 2\sigma_\lambda^2\end{bmatrix}=\begin{bmatrix}\sigma_{\lambda12}^2 & \sigma_{\lambda12\lambda31}\\ \sigma_{\lambda12\lambda31} & \sigma_{\lambda31}^2\end{bmatrix} \tag{5.58}$$

donde se conclui que

$$\begin{cases}\sigma_{\lambda12}=\sqrt{2}\sigma_\lambda\\ \sigma_{\lambda31}=\sqrt{2}\sigma_\lambda\\ \sigma_{\lambda12\lambda31}=-\sigma_\lambda\end{cases} \tag{5.59}$$

$$\rho_\lambda=\frac{\sigma_{\lambda12\lambda31}}{\sigma_{\lambda12}\cdot\sigma_{\lambda31}}=-0{,}5 \tag{5.60}$$

Partindo do princípio que λ_1, λ_2 e λ_3 possuem distribuição normal, a densidade de probabilidade conjunta de λ_{12} e λ_{31} é então dada por

$$f_{\lambda12\lambda31}(\lambda_{12},\lambda_{31})=\frac{1}{\sqrt{3}\pi\sigma_\lambda^2}e^{-\frac{m^2}{2}} \tag{5.61}$$

em que

$$m^2=\frac{2}{3\sigma_\lambda^2}\left[(\lambda_{12}-\lambda_{12m})^2+(\lambda_{12}-\lambda_{12m})(\lambda_{31}-\lambda_{31m})+(\lambda_{31}-\lambda_{31m})^2\right] \tag{5.62}$$

[42] Neste caso não há aproximações na determinação de $[\mathbf{C}_{\lambda12\lambda31}]$.

ou seja,

$$\frac{2}{3m^2\sigma_\lambda^2}\cdot\left[(\lambda_{12}-\lambda_{12m})^2+(\lambda_{12}-\lambda_{12m})(\lambda_{31}-\lambda_{31m})+(\lambda_{31}-\lambda_{31m})^2\right]=1 \tag{5.63}$$

Se m for constante, a expressão anterior é a equação de uma elipse Φ (Figura 5.85) situada no referencial $\lambda_{12}0\lambda_{31}$ e cujo centro de simetria é o ponto de coordenadas λ_{12m} e λ_{31m} (imagem inversa do ponto P_{Rcalc}). Os comprimentos dos seus semieixos são dados por (Anexo H)

$$\begin{cases} s_m = m\sigma_\lambda \\ s_M = \sqrt{3}m\sigma_\lambda \end{cases} \tag{5.64}$$

Esta elipse encontra-se inscrita numa superfície de incerteza de medição hexagonal cujos vértices são os seguintes pontos (Anexo H):

$$\begin{array}{ll} \mathbf{P1}\ (\lambda_{12m}-\sqrt{2}s_m\ ,\ \lambda_{31m}) & \mathbf{P1}\ (\lambda_{12m}-\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}) \\ \mathbf{P2}\ (\lambda_{12m}-\sqrt{2}s_m\ ,\ \lambda_{31m}+\sqrt{2}s_m) & \mathbf{P2}\ (\lambda_{12m}-\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}+\sqrt{2}m\sigma_\lambda) \\ \mathbf{P3}\ (\lambda_{12m}\ ,\ \lambda_{31m}+\sqrt{2}s_m) & \mathbf{P3}\ (\lambda_{12m}\ ,\ \lambda_{31m}+\sqrt{2}m\sigma_\lambda) \\ \mathbf{P4}\ (\lambda_{12m}+\sqrt{2}s_m\ ,\ \lambda_{31m}) & \mathbf{P4}\ (\lambda_{12m}+\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}) \\ \mathbf{P5}\ (\lambda_{12m}+\sqrt{2}e_m\ ,\ \lambda_{31m}-\sqrt{2}s_m) & \mathbf{P5}\ (\lambda_{12m}+\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}-\sqrt{2}m\sigma_\lambda) \\ \mathbf{P6}\ (\lambda_{12m}\ ,\ \lambda_{31m}-\sqrt{2}s_m) & \mathbf{P6}\ (\lambda_{12m}\ ,\ \lambda_{31m}-\sqrt{2}m\sigma_\lambda) \end{array} \tag{5.65}$$

Assim, a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro desta superfície de incerteza de medição é superior a p_λ e, por isso, a respectiva superfície de incerteza de posição possui uma probabilidade superior a p_λ de conter a posição verdadeira do robô, P_R.

A aplicação do algoritmo apresentado no ponto 5.6.2 para cálculo do erro máximo de posição a esta superfície de incerteza de posição produz um valor $\Delta P_{Rmáx}$ que, neste caso, tem o seguinte significado: *existe uma probabilidade superior a p_λ de que a distância entre a posição mais provável do robô P_{Rcalc} e a sua posição verdadeira P_R não ultrapasse o valor $\Delta P_{Rmáx}$.*

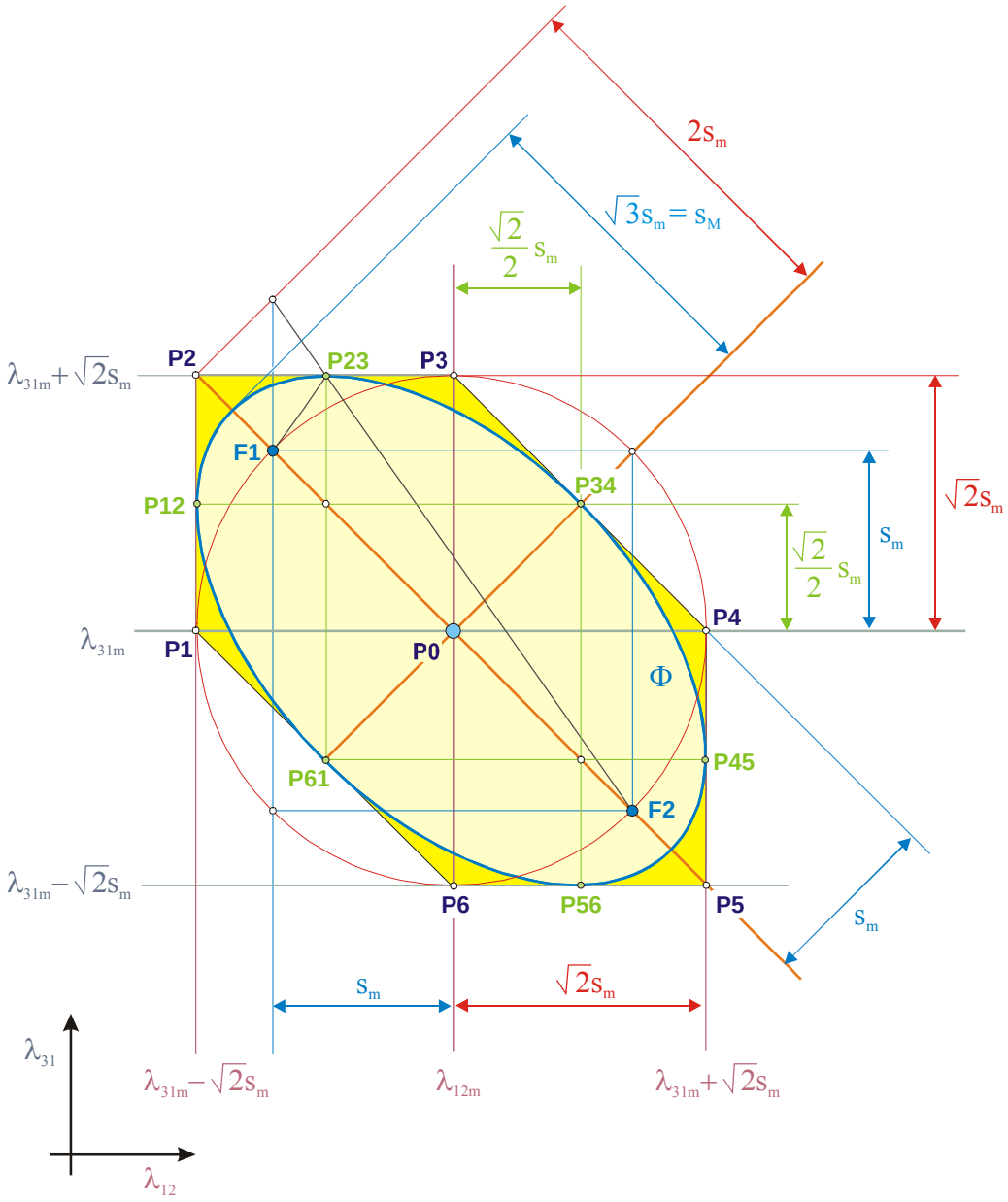


Figura 5.85: A probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são P1, P2, P3, P4, P5 e P6 é superior a p_λ, que é a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da elipse Φ.

Para m=1, a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da elipse Φ_{39}, representada na Figura 5.86, é de 39%:

$$m=1 \Rightarrow \begin{cases} p_\lambda = 1-e^{-\frac{1}{2}} = 0{,}39 \\ \dfrac{2}{3\sigma_\lambda^2}\cdot\left[(\lambda_{12}-\lambda_{12m})^2+(\lambda_{12}-\lambda_{12m})(\lambda_{31}-\lambda_{31m})+(\lambda_{31}-\lambda_{31m})^2\right]=1 \Rightarrow \begin{cases} s_m=\sigma_\lambda \\ s_M=\sqrt{3}\sigma_\lambda \end{cases} \end{cases} \quad (5.66)$$

Assim, é superior a 39% a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são os seguintes pontos:

$$\begin{aligned}
&\mathbf{P1_{39}}\ \left(\lambda_{12m}-\sqrt{2}\sigma_\lambda\ ,\ \lambda_{31m}\right)\\
&\mathbf{P2_{39}}\ \left(\lambda_{12m}-\sqrt{2}\sigma_\lambda\ ,\ \lambda_{31m}+\sqrt{2}\sigma_\lambda\right)\\
&\mathbf{P3_{39}}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}+\sqrt{2}\sigma_\lambda\right)\\
&\mathbf{P4_{39}}\ \left(\lambda_{12m}+\sqrt{2}\sigma_\lambda\ ,\ \lambda_{31m}\right)\\
&\mathbf{P5_{39}}\ \left(\lambda_{12m}+\sqrt{2}\sigma_\lambda\ ,\ \lambda_{31m}-\sqrt{2}\sigma_\lambda\right)\\
&\mathbf{P6_{39}}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}-\sqrt{2}\sigma_\lambda\right)
\end{aligned} \tag{5.67}$$

Uma vez calculado $\Delta P_{Rmáx}$ pode afirmar-se que existe uma probabilidade superior a 39% de que a distância entre a posição mais provável do robô P_{Rcalc} e a sua posição verdadeira P_R não ultrapasse o valor $\Delta P_{Rmáx}$.

A probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da elipse Φ_{95} (Figura 5.86) é de 95%, valor que se obtém fazendo m^2=6:

$$p_\lambda = 0{,}95 \Rightarrow \begin{cases} m^2 = -2\ln 0{,}05 \approx 6 \\ \dfrac{1}{9\sigma_\lambda^2}\cdot\left[(\lambda_{12}-\lambda_{12m})^2+(\lambda_{12}-\lambda_{12m})(\lambda_{31}-\lambda_{31m})+(\lambda_{31}-\lambda_{31m})^2\right]=1 \Rightarrow \begin{cases} s_m=\sqrt{6}\sigma_\lambda \\ s_M=3\sqrt{2}\sigma_\lambda \end{cases} \end{cases} \tag{5.68}$$

Neste caso, é superior a 95% a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são os seguintes pontos:

$$\begin{aligned}
&\mathbf{P1_{95}}\ \left(\lambda_{12m}-2\sqrt{3}\sigma_\lambda\ ,\ \lambda_{31m}\right)\\
&\mathbf{P2_{95}}\ \left(\lambda_{12m}-2\sqrt{3}\sigma_\lambda\ ,\ \lambda_{31m}+2\sqrt{3}\sigma_\lambda\right)\\
&\mathbf{P3_{95}}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}+2\sqrt{3}\sigma_\lambda\right)\\
&\mathbf{P4_{95}}\ \left(\lambda_{12m}+2\sqrt{3}\sigma_\lambda\ ,\ \lambda_{31m}\right)\\
&\mathbf{P5_{95}}\ \left(\lambda_{12m}+\sqrt{2}\sigma_\lambda\ ,\ \lambda_{31m}-\sqrt{2}\sigma_\lambda\right)\\
&\mathbf{P6_{95}}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}-\sqrt{2}\sigma_\lambda\right)
\end{aligned} \tag{5.69}$$

Então, uma vez calculado $\Delta P_{Rmáx}$ pode afirmar-se que existe uma probabilidade superior a 95% de que a distância entre a posição mais provável do robô P_{Rcalc} e a sua posição verdadeira P_R não ultrapasse o valor $\Delta P_{Rmáx}$.

À semelhança do que se fez no ponto 5.6.4, a seguir apresentam-se as condições que não se devem verificar a fim de garantir que, para cada posição calculada, todos os vértices da respectiva superfície de incerteza de medição possuem imagens no plano de navegação e nenhuma dessas imagens coincide com a posição de uma das três balizas utilizadas.

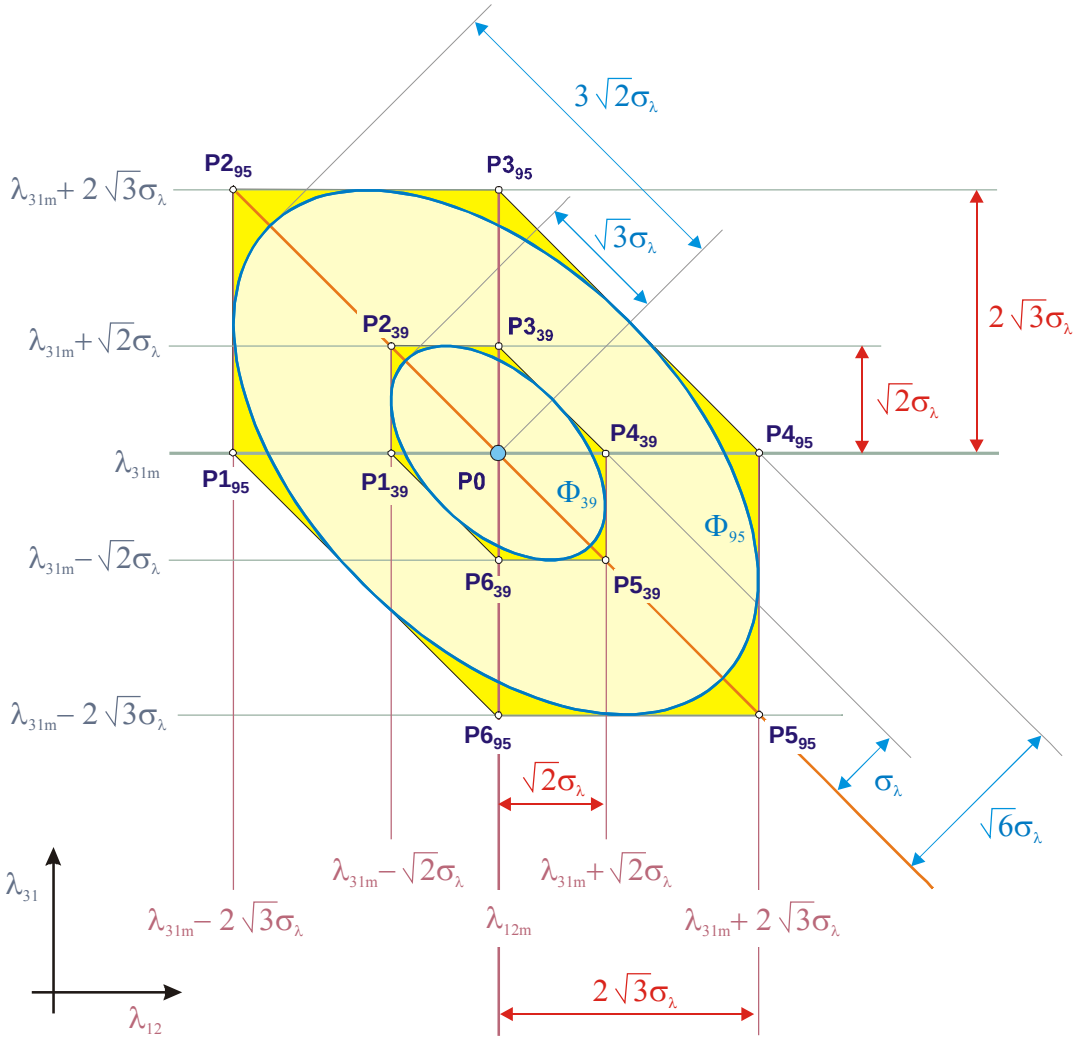


Figura 5.86: A probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são $P1_{39}$, $P2_{39}$, $P3_{39}$, $P4_{39}$, $P5_{39}$ e $P6_{39}$ é superior a 39%, que é a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da elipse Φ_{39}. E a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da superfície de incerteza de medição cujos vértices são $P1_{95}$, $P2_{95}$, $P3_{95}$, $P4_{95}$, $P5_{95}$ e $P6_{95}$ é superior a 95%, que é a probabilidade de os valores verdadeiros de λ_{12} e λ_{31} se situarem dentro da elipse Φ_{95}.

Com qualquer configuração de balizas, para ser possível efectuar a autolocalização e determinar $\Delta P_{Rmáx}$ quando a posição mais provável do robô no plano de navegação é a imagem do ponto (λ_{12m}, λ_{31m}) no referencial $\lambda_{12}0\lambda_{23}$, supondo que a cada um dos valores λ_{12m} e λ_{31m} está associada uma incerteza $\pm\sigma_\lambda$ e que λ_{12} e λ_{31} são estatisticamente independentes, é necessário e suficiente que não se verifique nenhuma das seguintes condições:

- $\left|\lambda_{12m} - (\sigma - \delta)\right| \leq m\sigma_\lambda$
- $\left|\lambda_{12m} - (\sigma - \delta + 360^o)\right| \leq m\sigma_\lambda$
- $\left|\lambda_{31m} - \delta\right| \leq m\sigma_\lambda$ (5.70)
- $\left|\lambda_{31m} - (\delta + 360^o)\right| \leq m\sigma_\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 180^o)\right| \leq 2m\sigma_\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 540^o)\right| \leq 2m\sigma_\lambda$

Com qualquer configuração de balizas, para ser possível efectuar a autolocalização e determinar $\Delta P_{Rmáx}$ quando a posição calculada do robô no plano de navegação é a imagem do ponto (λ_{12m}, λ_{31m}) no referencial $\lambda_{12}0\lambda_{23}$, supondo que λ_{12m} e λ_{31m} são calculados a partir dos valores λ_{1m}, λ_{2m} e λ_{3m}, a cada um dos quais está associada uma incerteza $\pm\sigma_\lambda$, e que λ_1, λ_2 e λ_3 são estatisticamente independentes, é necessário e suficiente que não se verifique nenhuma das seguintes condições:

- $\left|\lambda_{12m} - (\sigma - \delta)\right| \leq \sqrt{2}m\sigma_\lambda$
- $\left|\lambda_{12m} - (\sigma - \delta + 360^o)\right| \leq \sqrt{2}m\sigma_\lambda$
- $\left|\lambda_{31m} - \delta\right| \leq \sqrt{2}m\sigma_\lambda$ (5.71)
- $\left|\lambda_{31m} - (\delta + 360^o)\right| \leq \sqrt{2}m\sigma_\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 180^o)\right| \leq \sqrt{2}m\sigma_\lambda$
- $\left|(\lambda_{12m} + \lambda_{31m}) - (\sigma + 540^o)\right| \leq \sqrt{2}m\sigma_\lambda$

De acordo com a exposição feita, o algoritmo proposto no ponto 5.6.4 permite calcular uma distância tal que existe uma probabilidade superior a um valor previamente estipulado de o erro de posição devido aos erros aleatórios de medição não ser superior a essa distância. Não se efectuam aproximações e não se recorre ao cálculo de derivadas parciais com expressões analíticas difíceis de obter. Ao contrário do que acontece com os métodos que aproximam funções não lineares por funções lineares, não é necessário garantir que a incerteza de medição se mantém abaixo de um determinado limiar.

5.7 Conclusões

Apresentou-se um quadro de análise da autolocalização absoluta por triangulação com três balizas. Constitui a base sobre a qual se fará – no Capítulo 6 – a generalização do Algoritmo de Triangulação Geométrica, mas é aplicável a outros algoritmos de triangulação. Inclui um método capaz de determinar em tempo real as incertezas associadas à posição e à orientação calculadas, e de detectar situações nas quais a localização não é possível.

No ponto 5.1 definiram-se os ângulos σ, δ e ϕ, que caracterizam a configuração de balizas (estas formam um triângulo ou um segmento de recta que pode ser visto como um triângulo degenerado):

- σ e δ caracterizam a família de triângulos semelhantes à qual pertence o triângulo formado pelas balizas e também o sentido em que estas se encontram ordenadas;

- ϕ determina a orientação do triângulo formado pelas balizas relativamente aos eixos em que se medem as coordenadas no plano de navegação.

No ponto 5.2 definiram-se os ângulos λ_{12}, λ_{23} e λ_{31}, existentes entre os segmentos de recta que unem o robô a cada baliza. A análise realizada permite concluir que:

a) é possível determinar sem ambiguidade a posição do robô recorrendo a dois destes ângulos orientados, desde que o robô se encontre fora da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares.

b) o plano de navegação pode dividir-se em zonas bem determinadas, de acordo com os valores de λ_{12}, λ_{23} e λ_{31}, para as possíveis configurações de balizas.

c) em qualquer ponto do plano de navegação verifica-se sempre que $\lambda_{12}+\lambda_{23}+\lambda_{31}=360º$ ou então $\lambda_{12}+\lambda_{23}+\lambda_{31}=720º$.

d) em cada um dos três arcos em que se pode dividir a circunferência definida por três balizas não colineares, os ângulos λ_{12} e λ_{31} assumem valores particulares que só dependem de σ e δ e, portanto, são conhecidos *a priori*.

Isto constitui um meio de o algoritmo de triangulação detectar a presença do robô sobre a circunferência, podendo assim desencadear uma acção adequada a esta situação na qual a autolocalização não é possível.

e) a presença do robô sobre a recta que passa por três balizas colineares – situação na qual a autolocalização também não é possível – pode ser detectada pelo facto λ_{12} e λ_{31} assumirem valores 0º ou 180º.

No ponto 5.2 definiram-se também os ângulos λ_1, λ_2 e λ_3, existentes entre um semieixo de referência fixo no robô e os segmentos de recta que unem o robô a cada baliza.

No ponto 5.3 definiu-se o ângulo θ_R, que é a orientação do robô. Definiu-se também o ângulo τ que, em cada ponto, fica determinado pelos valores de λ_{12} e λ_{31} que aí ocorrem. O valor de θ_R nesse ponto pode ser calculado a partir de τ e de um dos ângulos λ_1, λ_2 ou λ_3.

No ponto 5.4 sugeriu-se uma nova especificação do problema da autolocalização absoluta a duas dimensões por triangulação, feita de acordo com as definições de ângulos propostas nos pontos anteriores.

No ponto 5.5 abordou-se a relação entre a posição do robô e os ângulos λ_{12} e λ_{31}, usados como variáveis de entrada em algoritmos de cálculo de posição por triangulação. Apresentaram-se, para todos os tipos de configurações de balizas, os gráficos que representam num referencial ortonormado $\lambda_{12}0\lambda_{31}$ os pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação. Salientaram-se algumas propriedades geométricas importantes destes gráficos.

No ponto 5.6 definiram-se o *erro de posição* ΔP_R e o *erro de orientação* $\Delta\theta_R$ existentes pelo facto de, em geral, a *posição calculada* mediante um determinado algoritmo de triangulação não coincidir com a *posição verdadeira* do robô e a *orientação calculada* também ser diferente da sua *orientação verdadeira*. Mostrou-se que é razoável admitir que estes erros se devem, sobretudo, aos erros de medição dos ângulos.

Quando é possível estabelecer valores máximos finitos para os erros de medição, as incertezas na posição e na orientação calculadas podem ser caracterizadas por um *erro máximo de posição* $\Delta P_{Rmáx}$ e um *erro máximo de orientação* $\Delta\theta_{Rmáx}$, valores máximos de ΔP_R e de $\Delta\theta_R$, respectivamente. Mostrou-se que:

a) Devido à incerteza de medição de ângulos não é possível determinar o ponto do plano de navegação que coincide com a posição verdadeira do robô mas, no caso de os erros de medição possuírem limites finitos e conhecidos, pode-se definir uma região do plano que contém esse ponto. Chamou-se *superfície de incerteza de posição* a essa região e *superfície de incerteza de medição* à sua imagem inversa no referencial $\lambda_{12}0\lambda_{31}$.

b) A mesma incerteza nas medições independentes de λ_{12} e λ_{31} realizadas em diversos pontos do plano de navegação produz superfícies de incerteza de posição cujas dimensões dependem do ponto em que as medições são realizadas.

c) O cálculo do erro máximo de posição recorrendo a aproximações de primeira ordem possui os seguintes inconvenientes:

   - O valor obtido para o erro máximo é aproximado.

   - Os cálculos envolvem derivadas parciais cujas expressões analíticas podem ser extremamente difíceis de obter, tornando-se necessário efectuar mais aproximações num método que já é aproximado por natureza.

   - Devido às aproximações inerentes ao método, este só é válido se a incerteza de medição de ângulos for suficientemente pequena.

d) O cálculo do erro máximo de posição a partir da área da superfície de incerteza de posição, recorrendo ao Jacobiano das funções utilizadas para determinar a posição do robô em função dos ângulos medidos, também possui as seguintes limitações:

   - O valor obtido para a área da superfície de incerteza de posição é aproximado.

- O Jacobiano pode ter de ser calculado a partir de derivadas parciais com expressões analíticas extremamente difíceis de obter. Mais uma vez se coloca o problema de ter de introduzir novas aproximações num método que já é aproximado pela sua própria natureza.

- Devido às aproximações inerentes ao método, este só é válido se a incerteza de medição de ângulos for suficientemente pequena.

- Não é garantido – pelo menos em princípio – que o erro de posição seja pequeno só pelo facto de a área da superfície de incerteza de posição ser pequena.

- O método pressupõe que as variáveis de entrada (λ_{12m} e λ_{31m}) são independentes, o que não é verdade se os seus valores forem calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3.

Sugeriu-se um novo método para calcular, em tempo real, os erros máximos de posição e de orientação ($\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$), partindo do princípio que os erros de medição têm limites finitos conhecidos. O método também permite detectar situações nas quais a localização não é possível. As suas características são as seguintes:

a) Funciona com todos os algoritmos de autolocalização absoluta por triangulação com três balizas (é independente do algoritmo de triangulação utilizado).

b) É exacto (não há aproximações que lhe sejam inerentes).

c) Por ser exacto, não requer que a incerteza de medição de ângulos se mantenha abaixo de um determinado limiar.

d) Não requer o cálculo de derivadas parciais com expressões analíticas difíceis de obter.

e) Pode utilizar-se quando λ_{12m} e λ_{31m} resultam de medições independentes dos ângulos λ_{12} e λ_{31} ou quando são calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3.

f) Só deve ser utilizado se o algoritmo de triangulação usado no cálculo da posição for suficientemente rápido uma vez que, para cada posição calculada do robô, esse algoritmo tem de ser executado

- 5 vezes se λ_{12m} e λ_{31m} resultarem de medições independentes dos ângulos λ_{12} e λ_{31};

- 7 vezes se λ_{12m} e λ_{31m} forem calculados a partir de medições independentes dos ângulos λ_1, λ_2 e λ_3.

g) Para que funcione correctamente, deve-se garantir que as coordenadas dos vértices da superfície de incerteza de medição são valores válidos de λ_{12} e λ_{31} e que cada vértice da superfície de incerteza de medição possui uma imagem no plano de navegação. É necessário e suficiente que não se verifiquem certas condições, que foram apresentadas. Isto implica uma redução da superfície navegável, que é tanto maior quanto maior for a incerteza de medição de ângulos.

Foi apresentado o modo de usar o algoritmo desenvolvido para determinar $\Delta P_{Rmáx}$ na caracterização da incerteza de posição devida aos erros aleatórios de medição, considerando que esses erros possuem distribuições de probabilidade gaussianas. O algoritmo permite calcular uma distância tal que existe uma probabilidade superior a um valor previamente estipulado de o erro de posição devido aos erros aleatórios de medição não ser superior a essa distância.

Como trabalho futuro, sugerem-se as seguintes tarefas:

- Investigar uma generalização da análise desenvolvida que seja adequada à autolocalização absoluta, a três dimensões, por triangulação.

- Estudar de forma mais aprofundada as propriedades geométricas dos gráficos que representam, num referencial ortonormado $\lambda_{12}0\lambda_{31}$, os pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação, para uma dada configuração de balizas, com o fim de obter uma mais completa caracterização dessa configuração.

- Estudar a relação entre τ e λ_1 tendo em vista produzir uma melhor estimativa do erro máximo de orientação.

- Investigar a possibilidade de se usar o algoritmo desenvolvido para determinar o erro máximo de orientação na caracterização da incerteza de orientação devida aos erros aleatórios de medição.

- Investigar as propriedades geométricas da superfície de incerteza de posição com a finalidade de descobrir se é possível simplificar os algoritmos apresentados neste capítulo.

- Elaborar um algoritmo capaz de escolher, entre todas as balizas visíveis de uma dada posição, o terno de balizas e a respectiva ordenação que permitem calcular a posição e a orientação do robô com valores mínimos de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. Uma solução óbvia é recorrer aos algoritmos apresentados neste capítulo e calcular os valores de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$ que se podem obter com cada um dos ternos de balizas disponíveis e com cada uma das ordenações possíveis, escolhendo depois o terno e a ordenação que geraram os menores valores de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. Mas talvez seja possível conhecer *a priori* (ou seja, antes de calcular a posição e a orientação) quais são, num dado ponto do plano de navegação, o terno de balizas e a ordenação que irão originar esses valores mínimos de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. Pode ser que parte da solução se encontre nas propriedades geométricas dos gráficos que representam, num referencial ortonormado $\lambda_{12}0\lambda_{31}$, os pontos correspondentes aos pares de valores $(\lambda_{12}, \lambda_{31})$ obtidos em cada posição do plano de navegação, para uma dada configuração de balizas.

- Desenvolver um método para determinar qual é a configuração de balizas que minimiza a fracção de um dado recinto de navegação na qual ocorrem erros de posição e de orientação superiores a um valor máximo admissível.

As soluções para os problemas referidos nas duas últimas tarefas propostas podem estar parcialmente radicadas no algoritmo de triangulação utilizado. Mas talvez seja possível chegar a algumas conclusões gerais, aplícáveis a todos os algoritmos de autolocalização absoluta por triangulação com três balizas.

6. Generalização do Algoritmo de Triangulação Geométrica

No Capítulo 4 apresentou-se o Algoritmo de Triangulação Geométrica, que está sujeito às restrições comuns a todos os outros algoritmos de autolocalização por triangulação.

Segundo Cohen e Koss (1992), as três balizas usadas por esse algoritmo também têm de ser "devidamente ordenadas".

Como resultado das restrições (tanto as gerais como as específicas), há zonas e percursos do plano de navegação nos quais o Algoritmo de Triangulação Geométrica não funciona.

Adicionalmente, Cohen e Koss (1992) advertem que *"o algoritmo só funciona consistentemente quando o robô se encontra dentro do triângulo formado pelas três balizas. Há zonas fora do triângulo de balizas nas quais o algoritmo funciona, mas essas zonas são difíceis de determinar e são altamente dependentes do modo como se definem os ângulos".*

Neste capítulo, no ponto 6.1, apresenta-se o Algoritmo Generalizado de Triangulação Geométrica (Sena Esteves *et al.*, 2003) que não requer ordenação de balizas e apenas está sujeito às restrições que são comuns a todos os algoritmos de autolocalização por triangulação (ponto 6.2).

As melhorias são conseguidas, sobretudo, graças a uma cuidadosa definição dos ângulos usados pelo algoritmo, que constitui uma solução para o problema referido por Cohen e Koss (1992) a propósito do Algoritmo de Triangulação Geométrica: "*Não foi encontrada uma regra consistente para definir os ângulos por forma a garantir uma correcta execução deste método"*.

No ponto 6.3 apresentam-se os resultados obtidos em quatro conjuntos de testes, realizados – mediante a simulação por computador – com as seguintes finalidades:

- Validar o Algoritmo Generalizado de Triangulação Geométrica;

- Verificar de que modo é que os erros de medição dos ângulos afectam os erros de posição e de orientação, quando o robô se encontra próximo ou afastado das balizas;

- Validar o método de caracterização das incertezas de posição e de orientação que foi proposto no Capítulo 5;

- Determinar, para várias posições do robô relativamente às balizas, o tempo necessário para calcular a posição, a orientação e as incertezas que lhes estão associadas, quando se recorre ao Algoritmo Generalizado de Triangulação Geométrica e ao método de caracterização das incertezas de posição e de orientação proposto no capítulo 5.

O capítulo termina com as conclusões apresentadas no ponto 6.4.

6.1 O Algoritmo

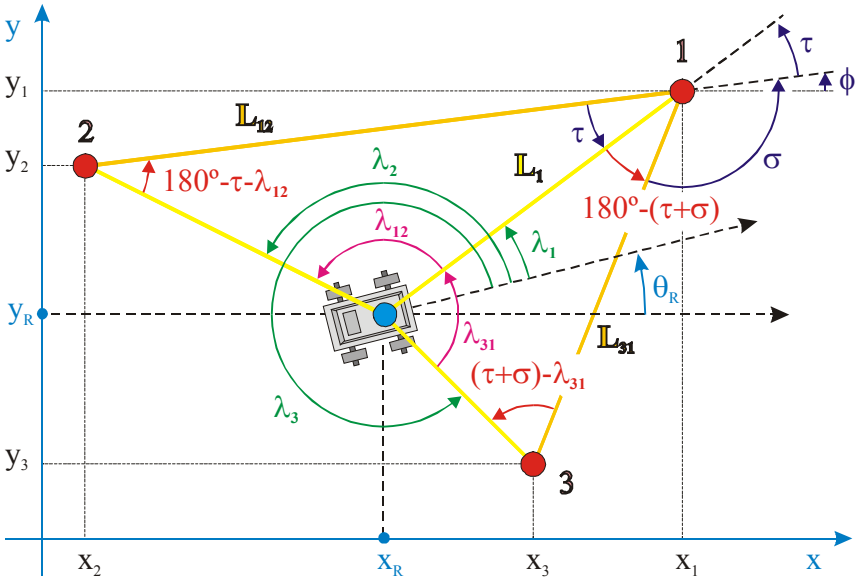
Considerem-se (Figura 6.1) três balizas[1] numeradas aleatoriamente de 1 a 3, situadas em posições conhecidas de um plano de navegação no qual se definiu um referencial ortonormado x0y. As coordenadas dos pontos nos quais se situam as balizas 1, 2 e 3 são, respectivamente, (x_1, y_1), (x_2, y_2) e (x_3, y_3). L_{12} e L_{31} são, respectivamente, as distâncias entre as balizas 1 e 2 e as balizas 1 e 3.

Para determinar a sua posição (x_R, y_R) e a sua orientação θ_R, o robô[2] mede os ângulos[3] λ_1, λ_2 e λ_3 (definidos no Capítulo 5). L_1 é a distância entre o robô e a baliza 1. As linhas 2 a 5 do algoritmo calculam os ângulos λ_{12} e λ_{31} (definidos no Capítulo 5).

[1] Parte-se do princípio que todas as balizas referidas neste capítulo são emissoras omnidireccionais pontuais e distinguíveis, com padrão de emissão isotrópico e alcance infinito.

[2] A posição verdadeira do robô é a posição de um dos seus pontos. Neste capítulo usa-se esse ponto para representar todo o robô, cujas dimensões não são relevantes para a análise realizada. Por simplicidade de linguagem, chama-se *"o robô"* ao ponto. Isto deve ser tido em consideração quando se faz referência à *"distância entre o robô e uma das balizas"* ou se indica que *"o robô se encontra sobre a circunferência definida por três balizas não colineares",* por exemplo.

[3] Parte-se do princípio que estes ângulos são medidos a partir do mesmo ponto do plano de navegação e que o robô não muda a sua orientação enquanto as medições estão a ser feitas.



Limites dos Ângulos

$0º \le \lambda_1 < 360º$
$0º \le \lambda_2 < 360º$
$0º \le \lambda_3 < 360º$

$0º < \lambda_{12} < 360º$
$0º < \lambda_{31} < 360º$

$-180º < \phi \le 180º$
$-180º < \sigma \le 180º$
$-180º < \tau \le 180º$

$-180º < \theta_R \le 180º$

1. Se houver menos de três balizas à vista, emitir uma mensagem de erro e parar.
2. $\lambda_{12} = \lambda_2 - \lambda_1$
3. Se $\lambda_1 > \lambda_2$ então $\lambda_{12} = \lambda_{12} + 360º$
4. $\lambda_{31} = \lambda_1 - \lambda_3$
5. Se $\lambda_3 > \lambda_1$ então $\lambda_{31} = \lambda_{31} + 360º$
6. Calcular L_{12} a partir das posições conhecidas das balizas 1 e 2.
7. Calcular L_{31} a partir das posições conhecidas das balizas 1 e 3.
8. Seja ϕ um ângulo orientado tal que $-180º < \phi \le 180º$. O seu lado origem é a imagem do semieixo positivo dos xx que resulta da translação associada ao vector cuja origem é a origem do referencial e termina na baliza 1. O lado extremidade é a parte da recta que passa pelas balizas 1 e 2 cuja origem é a baliza 1 e não contém a baliza 2.
9. Seja σ um ângulo orientado tal que $-180º < \sigma \le 180º$. O seu lado origem é o segmento de recta que une as balizas 1 e 3. O lado extremidade é a parte da recta que passa pelas balizas 1 e 2 cuja origem é a baliza 1 e não contém a baliza 2.
10. $\gamma = \sigma - \lambda_{31}$
11. $\tau = \text{arctg}\left[\dfrac{\text{sen}\lambda_{12} \cdot (L_{12} \cdot \text{sen}\lambda_{31} - L_{31} \cdot \text{sen}\gamma)}{L_{31} \cdot \text{sen}\lambda_{12} \cdot \cos\gamma - L_{12} \cdot \cos\lambda_{12} \cdot \text{sen}\lambda_{31}}\right]$
12. Se $\begin{cases}\lambda_{12} < 180º \\ \tau < 0º\end{cases}$ então $\tau = \tau + 180º$
13. Se $\begin{cases}\lambda_{12} > 180º \\ \tau > 0º\end{cases}$ então $\tau = \tau - 180º$
14. Se $\tau = 0º \ \wedge \left[\begin{cases}\sigma > 0º \\ \lambda_{31} > 180º\end{cases} \vee \begin{cases}\sigma < 0º \\ \lambda_{31} < 180º\end{cases}\right]$ então $\tau = 180º$
15. Se $|\text{sen}\lambda_{12}| > |\text{sen}\lambda_{31}|$ então $L_1 = \dfrac{L_{12} \cdot \text{sen}(\tau + \lambda_{12})}{\text{sen}\lambda_{12}}$
16. senão $L_1 = \dfrac{L_{31} \cdot \text{sen}(\tau + \sigma - \lambda_{31})}{\text{sen}\lambda_{31}}$
17. $x_R = x_1 - L_1 \cdot \cos(\phi + \tau)$
18. $y_R = y_1 - L_1 \cdot \text{sen}(\phi + \tau)$
19. $\theta_R = \phi + \tau - \lambda_1$
20. Se $\theta_R \le -180º$ então $\theta_R = \theta_R + 360º$
21. Se $\theta_R > 180º$ então $\theta_R = \theta_R - 360º$

Figura 6.1: Algoritmo Generalizado de Triangulação Geométrica.

O algoritmo utiliza as definições sugeridas no Capítulo 5 para os ângulos σ e ϕ:

- *Seja σ um ângulo orientado tal que -180º < σ ≤ 180º. O seu lado origem é o segmento de recta que une as balizas 1 e 3. O lado extremidade é a parte da recta que passa pelas balizas 1 e 2 cuja origem é a baliza 1 e não contém a baliza 2.*

- *Seja ϕ um ângulo orientado tal que -180º < ϕ ≤ 180º. O seu lado origem é a imagem do semieixo positivo dos xx que resulta da translação associada ao vector cuja origem é a origem do referencial x0y e termina na baliza 1. O lado extremidade é a parte da recta que passa pelas balizas 1 e 2 cuja origem é a baliza 1 e não contém a baliza 2.*

O ângulo τ está implicitamente definido pelo modo como é calculado. A sua definição explícita é a sugerida no Capítulo 5:

- *Seja τ um ângulo orientado tal que -180º < τ ≤ 180º. O seu lado origem é o segmento de recta que une as balizas 1 e 2. O lado extremidade é o segmento de recta que une o robô e a baliza 1.*

Como se viu no Capítulo 5, para cada um dos cinco tipos de configuração de balizas, de acordo com λ_{12} e λ_{31} é possível dividir o plano de navegação nas zonas apresentadas na Figura 6.2 (indica-se o sinal de τ em cada uma dessas zonas).

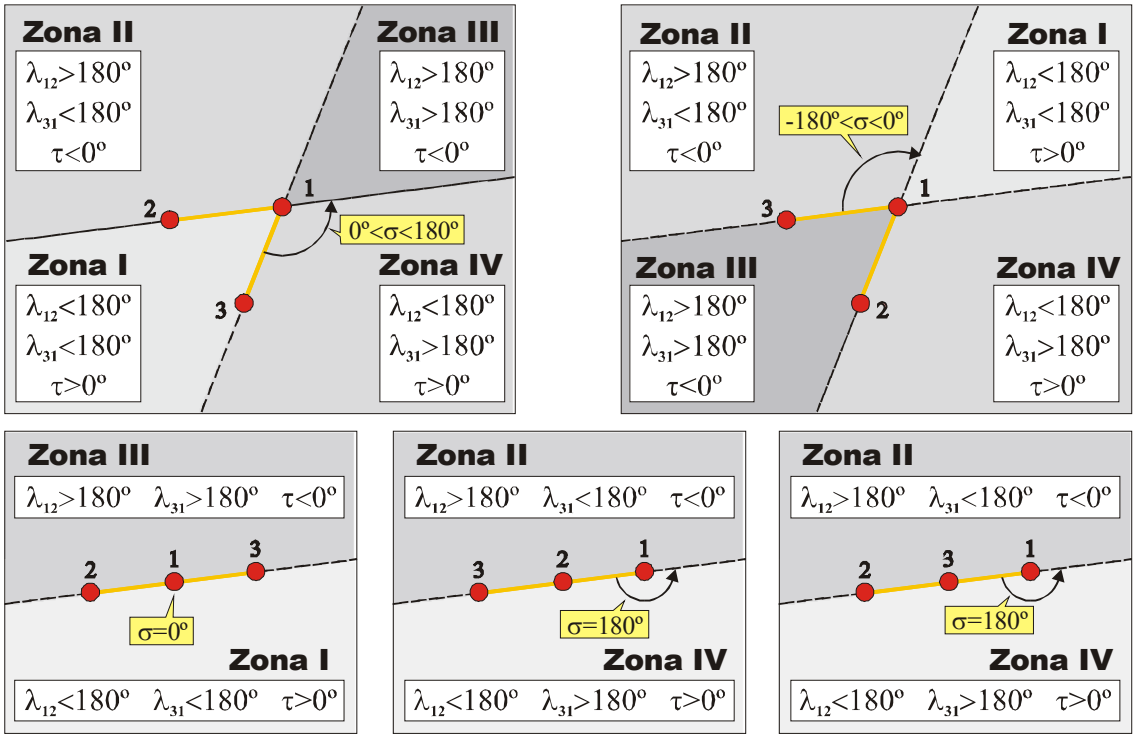


Figura 6.2: Divisão do plano de navegação, de acordo com os valores de λ_{12} e λ_{31}.

Aplicando a lei dos senos aos triângulos formados pelo robô e as balizas em cada zona do plano de navegação, obtêm-se as seguintes expressões (para a Zona I e $0º<\sigma<180º$, pode recorrer-se à situação representada na Figura 6.1):

Zona I:
$$\begin{cases} \dfrac{L_{31}}{\operatorname{sen}\lambda_{31}} = \dfrac{L_1}{\operatorname{sen}(\tau+\sigma-\lambda_{31})} \\ \dfrac{L_{12}}{\operatorname{sen}\lambda_{12}} = \dfrac{L_1}{\operatorname{sen}(180º-\tau-\lambda_{12})} \end{cases} \tag{6.1}$$

Zona II:
$$\begin{cases} \dfrac{L_{31}}{\operatorname{sen}\lambda_{31}} = \dfrac{L_1}{\operatorname{sen}(\tau+\sigma-\lambda_{31})} \\ \dfrac{L_{12}}{\operatorname{sen}(360º-\lambda_{12})} = \dfrac{L_1}{\operatorname{sen}(-180º+\tau+\lambda_{12})} \end{cases} \tag{6.2}$$

Zona III:
$$\begin{cases} \dfrac{L_{31}}{\operatorname{sen}(360º-\lambda_{31})} = \dfrac{L_1}{\operatorname{sen}(\lambda_{31}-\tau-\sigma)} \\ \dfrac{L_{12}}{\operatorname{sen}\lambda_{12}} = \dfrac{L_1}{\operatorname{sen}(180º-\tau-\lambda_{12})} \end{cases} \tag{6.3}$$

Zona IV:
$$\begin{cases} \dfrac{L_{31}}{\operatorname{sen}(360º-\lambda_{31})} = \dfrac{L_1}{\operatorname{sen}(\lambda_{31}-(\tau+\sigma)-360º)} \\ \dfrac{L_{12}}{\operatorname{sen}(360º-\lambda_{12})} = \dfrac{L_1}{\operatorname{sen}(-180º+\tau+\lambda_{12})} \end{cases} \tag{6.4}$$

Uma vez que, por definição,

$$\gamma = \sigma - \lambda_{31}, \tag{6.5}$$

resolvendo (6.1), (6.2), (6.3) e (6.4) para obter τ e L_1, obtém-se o mesmo resultado em cada zona:

$$\tau = \operatorname{arctg}\left[\frac{\operatorname{sen}\lambda_{12}\cdot(L_{12}\cdot\operatorname{sen}\lambda_{31}-L_{31}\cdot\operatorname{sen}\gamma)}{L_{31}\cdot\operatorname{sen}\lambda_{12}\cdot\cos\gamma-L_{12}\cdot\cos\lambda_{12}\cdot\operatorname{sen}\lambda_{31}}\right] \tag{6.6}$$

$$L_1 = \frac{L_{12}\cdot\operatorname{sen}(\tau+\lambda_{12})}{\operatorname{sen}\lambda_{12}} \qquad (\text{válido se } \operatorname{sen}\lambda_{12}\neq 0) \tag{6.7}$$

$$L_1 = \frac{L_{31}\cdot\operatorname{sen}(\tau+\sigma-\lambda_{31})}{\operatorname{sen}\lambda_{31}} \qquad (\text{válido se } \operatorname{sen}\lambda_{31}\neq 0) \tag{6.8}$$

Com três balizas não colineares, para possibilitar a localização se λ_{12}=180º ou λ_{31}=180º (Figura 6.3 e Figura 6.4), no cálculo de L_1 o algoritmo escolhe (linhas 14 e 15), entre as expressões (6.7) e (6.8), a que tiver maior denominador. Se λ_{12}=180º então o robô está sobre o segmento de recta que une as balizas 1 e 2 (Figura 6.3), e verifica-se que

$$L_1 = \frac{L_{31} \cdot \text{sen}(\sigma - \lambda_{31})}{\text{sen}\lambda_{31}}. \qquad (6.9)$$

Uma vez que τ=0º quando λ_{12}=180º, para calcular L_1 nessa circunstância é possível utilizar a expressão (6.8) em vez da expressão (6.9).

De igual modo, se λ_{31}=180º então o robô está sobre o segmento de recta que une as balizas 1 e 3 (Figura 6.4) e a expressão (6.7) pode ser usada para calcular L_1, se as três balizas forem não colineares.

Com três balizas colineares, se λ_{12}=180º ou λ_{31}=180º então o robô está sobre a recta definida pelas balizas e L_1 não se pode calcular.

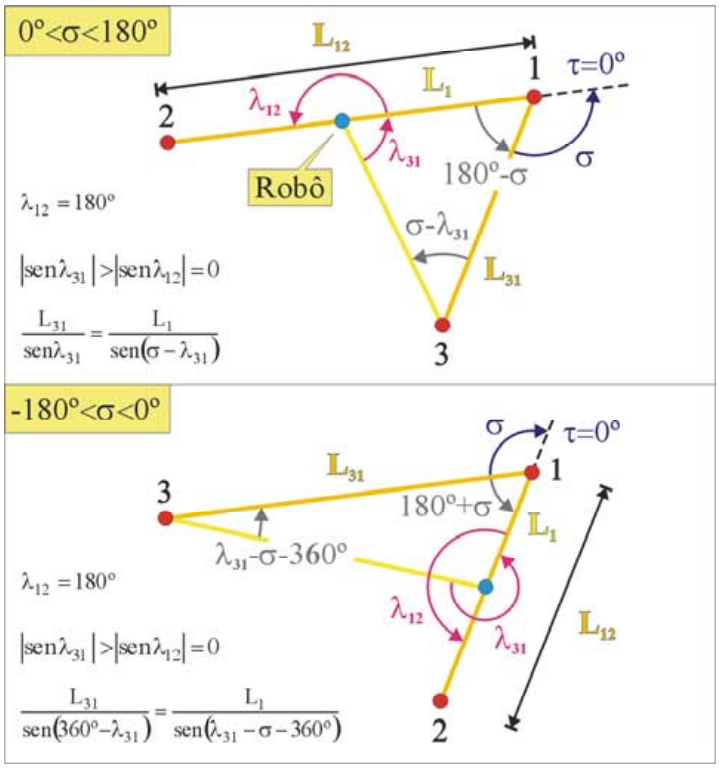


Figura 6.3: Se λ_{12}=180º então o robô está sobre o segmento de recta que une as balizas 1 e 2.

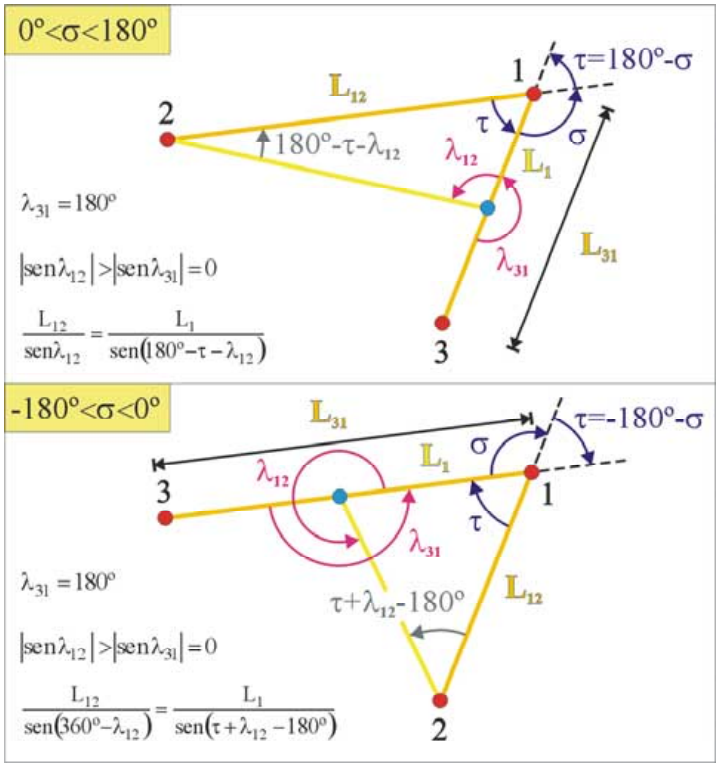


Figura 6.4: Se λ_{31}=180º então o robô está sobre o segmento de recta que une as balizas 1 e 3.

A função arctg devolve um valor que pertence ao intervalo compreendido entre -90º e 90º. Assim, as linhas 12, 13 e 14 do algoritmo são necessárias para calcular valores de τ que se encontrem fora desse intervalo. Em particular, estas linhas garantem o correcto funcionamento do algoritmo quando λ_{12}=0º ou λ_{31}=0º, ou seja, quando o robô se encontra sobre a recta definida pelas balizas 1 e 2, e fora do segmento de recta que une essas duas balizas, ou sobre a recta definida pelas balizas 2 e 3, e fora do segmento de recta que une essas duas balizas. A linha 14 só é necessária[4] quando o robô se encontra sobre a parte da recta que passa pelas balizas 1 e 2 cuja origem é a baliza 1 e não contém a baliza 2. Nesse caso, τ = 180º mas a expressão (6.6) dá um valor zero.

As linhas 20 e 21 do algoritmo garantem que -180º<θ ≤180º e podem ser omitidas se θ puder assumir valores fora desse intervalo.

[4] Na versão original do Algoritmo Generalizado de Triangulação Geométrica (Sena Esteves *et al.*, 2003) não aparece esta linha pelo facto de se ter partido do princípio que, quando uma das balizas se encontra sobre o segmento de recta que une o robô a outra baliza, a baliza mais afastada do robô fica ocultada pela mais próxima ou então o goniómetro não tem a capacidade de detectar simultaneamente duas balizas que sejam "vistas" na mesma direcção (definindo um enfiamento). Qualquer uma dessas situações impede a autolocalização. No entanto, o impedimento fica a dever-se à tecnologia utilizada e não ao algoritmo em si.

6.2 Restrições Específicas do Algoritmo

Como qualquer outro algoritmo de autolocalização por triangulação, o Algoritmo Generalizado de Triangulação Geométrica está sujeito a duas restrições:

- é necessário que haja um mínimo de três balizas visíveis;
- a localização não pode ser feita se o robô se encontrar sobre a circunferência definida por três balizas não colineares (Figura 6.5) ou sobre a recta definida por três balizas colineares (Figura 6.6).

No Algoritmo Generalizado de Triangulação Geométrica, a segunda restrição indicada aparece como uma impossibilidade de calcular τ por causa de uma indeterminação do tipo 0/0 na expressão (6.6). Esta indeterminação, se o robô estiver sobre a recta definida por três balizas colineares, deve-se ao facto de cada um do ângulos λ_{12} e λ_{31} valer[5] 0º ou 180º. Assim, os senos desses ângulos são sempre nulos.

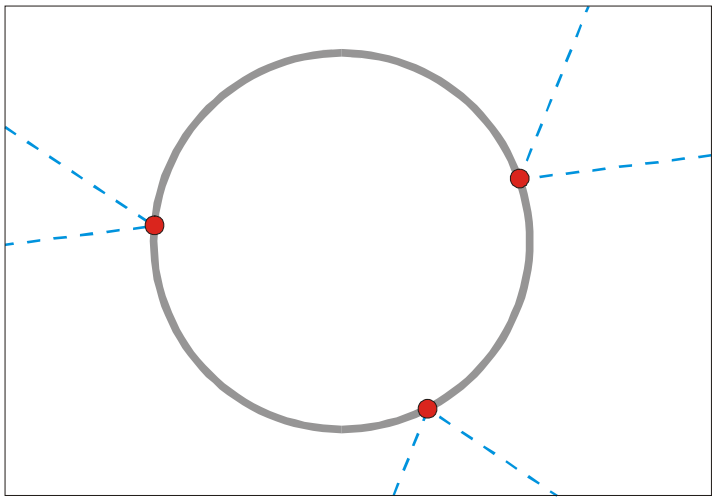


Figura 6.5: A localização não pode ser feita se o robô se encontrar sobre a circunferência definida por três balizas não colineares. Se, quando uma das balizas se encontra sobre o segmento de recta que une o robô a outra baliza, a baliza mais afastada do robô for ocultada pela mais próxima ou então o goniómetro não tiver a capacidade de detectar simultaneamente as duas balizas, a autolocalização também não é possível sobre os segmentos de recta representados a tracejado. Esta restrição adicional, que resulta da necessidade de haver três balizas visíveis, deve-se à tecnologia utilizada e não ao algoritmo em si.

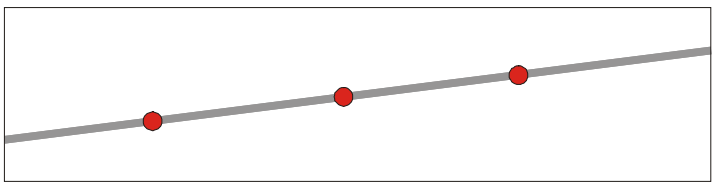


Figura 6.6: A localização não pode ser feita se o robô se encontrar sobre a recta definida por três balizas colineares.

[5] Se os ângulos medidos estiverem isentos de erro e não forem cometidos erros de cálculo.

Quando o robô está sobre a circunferência definida por três balizas não colineares, λ_{12} e λ_{31} assumem os valores indicados[6], para cada arco, na Figura 6.7 e na Figura 6.8. A análise destas figuras permite também concluir que

$$L_{12} \cdot \text{sen}\delta = L_{31} \cdot \text{sen}(\sigma - \delta) \ . \qquad (6.10)$$

Cada um dos seis pares de valores de λ_{12} e λ_{31} que ocorrem sobre os três arcos de cada circunferência, tendo em consideração (6.10), origina sempre o mesmo resultado, que causa uma indeterminação do tipo 0/0 na expressão (6.6):

$$\begin{cases} \text{sen}\lambda_{12} \cdot \left[L_{12} \cdot \text{sen}\lambda_{31} - L_{31} \cdot \text{sen}(\sigma - \lambda_{31}) \right] = 0 \\ L_{31} \cdot \text{sen}\lambda_{12} \cdot \cos(\sigma - \lambda_{31}) - L_{12} \cdot \cos\lambda_{12} \cdot \text{sen}\lambda_{31} = 0 \end{cases} \qquad (6.11)$$

Mesmo desprezando os erros de cálculo, esta indeterminação pode não chegar a ocorrer por causa dos erros de medição dos ângulos[7]. No entanto, quando o robô estiver sobre ou próximo da recta definida por três balizas colineares ou da circunferência definida por três balizas não colineares, é de esperar que os erros no valor calculado de τ produzam significativos erros de posição e de orientação.

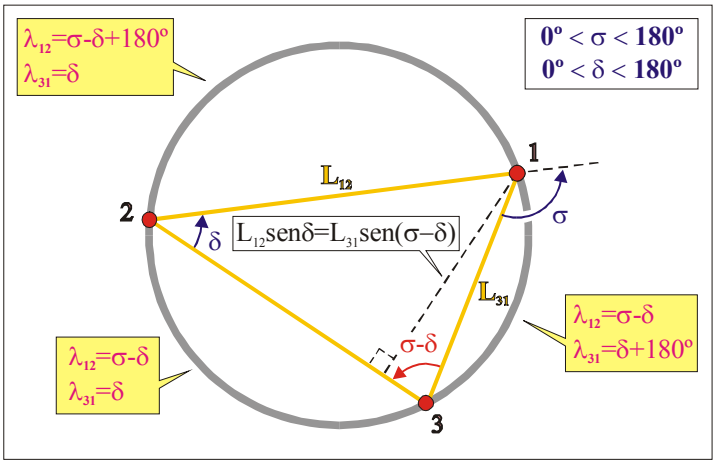


Figura 6.7: λ_{12} e λ_{31} sobre a circunferência definida por três balizas não colineares ordenadas no sentido directo.

[6] Se os ângulos medidos estiverem isentos de erro e não forem cometidos erros de cálculo.

[7] Devido a estes erros, quando o robô se encontra sobre a recta definida por três balizas colineares, λ_{12} e λ_{31} podem assumir valores diferentes de 0º ou 180º. E quando o robô se encontra sobre circunferência definida por três balizas não colineares, λ_{12} e λ_{31} podem assumir valores diferentes dos indicados na Figura 6.7 e na Figura 6.8.

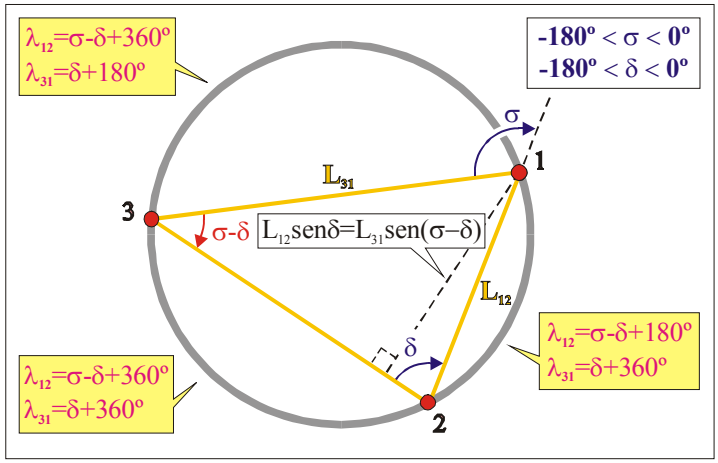


Figura 6.8: λ_{12} e λ_{31} sobre a circunferência definida por três balizas não colineares ordenadas no sentido inverso.

O Algoritmo Generalizado de Triangulação Geométrica está sujeito às restrições que são comuns a todos os algoritmos de autolocalização por triangulação, mas as restrições específicas do Algoritmo de Triangulação Geométrica não se lhe aplicam. De facto, o algoritmo generalizado possui as seguintes características:

- As três balizas usadas pelo algoritmo podem ser aleatoriamente numeradas 1, 2 e 3 (não é necessário numerar as balizas consecutivamente no sentido directo).

- As três balizas podem ser colocadas em quaisquer pontos do plano de navegação (desde que duas balizas não partilhem a mesma posição).

- O ângulo formado pelos segmentos de recta que unem o robô às balizas 1 e 2 (λ_{12}) e o ângulo formado pelos segmentos de recta que unem o robô às balizas 1 e 2 (λ_{31}) podem ser iguais ou superiores a 180º. Em particular, é possível a autolocalização do robô quando este se encontra sobre a recta definida pelas balizas 1 e 2 ou a recta definida pelas balizas 1 e 3.

- O algoritmo funciona de forma consistente dentro, fora ou sobre o triângulo formado por três balizas não colineares (excepto nos pontos em que se aplica alguma das restrições comuns a todos os algoritmos de autolocalização por triangulação) ou fora da recta definida por três balizas colineares.

6.3 Resultados Obtidos Mediante Simulação por Computador

Neste ponto apresentam-se os resultados obtidos em quatro conjuntos de testes realizados mediante a simulação por computador:

- Cinco testes para validar o Algoritmo Generalizado de Triangulação Geométrica, verificando o seu funcionamento com cada um dos cinco tipos de configurações de balizas representados na Figura 6.2;

- Seis testes para verificar de que modo é que os erros de medição dos ângulos afectam os erros de posição e de orientação, quando um robô se encontra próximo ou afastado das balizas;

- Seis testes para validar o método de caracterização das incertezas de posição e de orientação que foi proposto no Capítulo 5;

- Dez testes para determinar, em várias posições de um robô relativamente a três balizas, o tempo necessário para calcular a sua posição, a sua orientação e as incertezas que lhes estão associadas, quando se recorre ao Algoritmo Generalizado de Triangulação Geométrica e ao método de caracterização das incertezas de posição e de orientação proposto no Capítulo 5.

Em cada um destes testes, um robô virtual pontual navega num plano no qual se definiu um referencial ortonormado x0y. Três balizas virtuais, pontuais e distinguíveis, situadas em pontos conhecidos desse plano, tornam possível a autolocalização do robô recorrendo ao Algoritmo Generalizado de Triangulação Geométrica.

O código fonte dos programas de teste (Anexo I) foi escrito em *Java 2,* versão de 1998 da linguagem de programação *Java*, desenvolvida pela *Sun Microsystems.* Teria sido possível recorrer a outras linguagens de uso geral, por exemplo ao *C* ou ao *C++*. Optou-se pelo *Java*, sobretudo[8], pelo facto de ser mais simples de aprender e utilizar que o *C* ou o *C++* (pelo menos, ao nível exigido para a realização dos testes em questão) e também porque permite escrever programas melhores[9], mais claros e mais

[8] Para o trabalho realizado não foram relevantes algumas das características mais importantes do *Java*, por exemplo o facto de ser uma linguagem orientada aos objectos.

[9] O *Java* promove boas práticas de programação.

legíveis que os escritos nessas linguagens (Campione *et al.,* 2001; Martins, 2001). Também foi determinante o facto de a *Sun Microsystems* disponibilizar gratuitamente, via *Internet,* os seguintes elementos:

- Abundante informação sobre *Java 2;*
- Um ambiente de desenvolvimento de programas nesta linguagem, designado *Software Development Kit (SDK);*
- O pacote de *software Java 3D,* que contém vários métodos muito úteis para realizar operações com vectores.

Um programa escrito em *Java* pode ser executado de forma consistente em qualquer plataforma *Java*. Para isto ser possível, o *código fonte* de cada programa é convertido – graças a um compilador – num ficheiro de *bytecodes,* que são códigos independentes da máquina a utilizar. Posteriormente, de cada vez que o programa é executado numa máquina, os *bytecodes* são convertidos em *código nativo* dessa máquina por um interpretador de *bytecodes* que faz parte da chamada *Java Virtual Machine (JVM)*. Esta é um *motor de execução de software* que executa os *bytecodes* num microprocesador (que pode ser de um computador ou de outro dispositivo electrónico). Cada sistema operativo requer uma *JVM* específica, mas cada ficheiro de *bytecodes* pode ser executado em qualquer máquina que possua uma *JVM*.

De acordo com o exposto no parágrafo anterior, o *Java* pode considerar-se uma linguagem de programação simultaneamente compilada e interpretada embora, em termos de execução final, seja efectivamente interpretada (Campione *et al.,* 2001; Martins, 2001). Por isso, a performance dos programas escritos nesta linguagem não se pode comparar à que é possível obter com linguagens compiladas (por exemplo *C*), para certas aplicações (Martins, 2001). Com a *JVM* que foi usada para executar os programas de teste, a performance pode ser significativamente melhorada, em alguns casos, graças a um *compilador adaptativo* cuja utilização se descreve no ponto 6.3.4.

Para desenvolver os programas de teste recorreu-se à versão 1.3 para Windows do *Java 2 SDK, Standard Edition,* actualizada com o pacote *Java 3D* (versão 1.2.1 Beta, para *Win32/DirectX*), em ambiente *Windows XP* (versão 5.1.2600), num computador

pessoal equipado com um processador *Intel Pentium III* a funcionar a uma frequência de *clock* de 995MHz. Salvo indicação contrária, nos cálculos realizados usou-se o formato em vírgula flutuante IEEE 754 *Double* (64 bits).

Os gráficos que permitem visualizar os resultados obtidos em cada teste foram elaborados com o programa *Matlab* (versão 5.2), escolhido por produzir gráficos de excelente qualidade e por ser relativamente fácil de utilizar.

6.3.1 Primeiro Conjunto de Testes

O primeiro conjunto de cinco testes destina-se a validar o Algoritmo Generalizado de Triangulação Geométrica, verificando o seu funcionamento com cada um dos cinco tipos de configurações de balizas representados na Figura 6.2 e que correspondem a:

- balizas não colineares ordenadas no sentido directo;
- balizas não colineares ordenadas no sentido inverso;
- balizas colineares, estando a baliza 1 entre as balizas 2 e 3;
- balizas colineares, estando a baliza 3 entre as balizas 1 e 2;
- balizas colineares, estando a baliza 2 entre as balizas 1 e 3.

Em cada teste, três balizas numeradas 1, 2 e 3 são colocadas em posições conhecidas de um plano no qual está definido um referencial ortonormado x0y. As posições das balizas e os correspondentes valores de σ e ϕ estão indicadas junto dos resultados de cada teste. Coloca-se um robô pontual na origem do referencial e atribui-se à sua orientação um valor aleatoriamente escolhido, superior a -180º e inferior ou igual a 180º. De seguida, realiza-se uma sequência de quatro passos (Figura 6.9).

I. Os ângulos λ_1, λ_2 e λ_3 são calculados a partir das posições – conhecidas *a priori* – das balizas e do robô.

II. Os ângulos λ_1, λ_2 e λ_3 são arredondados para números inteiros. Com esta operação simula-se a saída de um goniómetro digital com uma resolução ρ igual a 1º. A correspondente incerteza de medição dos ângulos ($\pm\Delta\lambda$) é de ±0,5º.

III. O Algoritmo Generalizado de Triangulação Geométrica calcula a posição e a orientação do robô a partir dos valores arredondados de λ_1, λ_2 e λ_3 e das posições – conhecidas *a priori* – das balizas.

IV. Os erros de posição e de orientação são calculados comparando a posição e a orientação do robô conhecidas *a priori* com a posição a orientação do robô determinadas pelo Algoritmo Generalizado de Triangulação Geométrica.

Repetem-se os quatro passos para posições do robô cobrindo um quadrado de 100 × 100 unidades de comprimento, com incrementos de 0,1 unidades de comprimento tanto na direcção *x* como na direcção *y*. Em cada ponto, atribui-se à orientação do robô um valor aleatoriamente escolhido, superior a -180º e inferior ou igual a 180º.

Os erros de posição e de orientação obtidos em cada ponto encontram-se representados, para cada configuração de balizas, em gráficos bidimensionais (Figura 6.10) usando uma escala em tons de cinzento tal que um ponto se torna mais escuro à medida que aumenta o erro correspondente. Os eixos dos gráficos relativos aos erros de posição estão graduados nas mesmas unidades de comprimento. Os eixos dos *zz* dos gráficos relativos aos erros de orientação estão graduados em graus.

Os resultados apresentados estão de acordo com a análise realizada anteriormente. O Algoritmo Generalizado de Triangulação Geométrica funciona com cada um dos cinco tipos de configurações de balizas representados na Figura 6.2. Como se esperava, os erros de posição e de orientação são significativos quando o robô está sobre ou perto da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares.

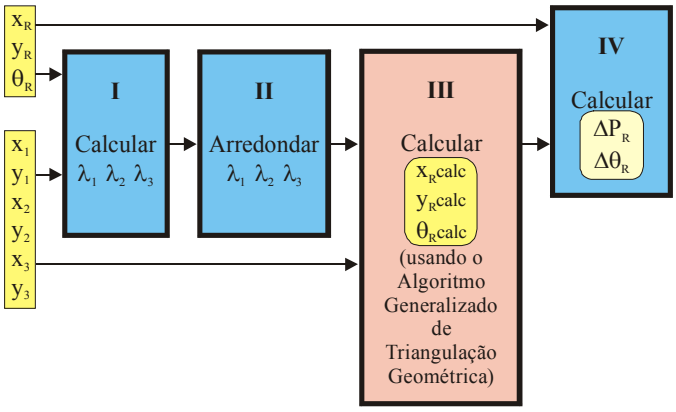


Figura 6.9: Sequência de passos executados no primeiro e no segundo conjunto de testes.

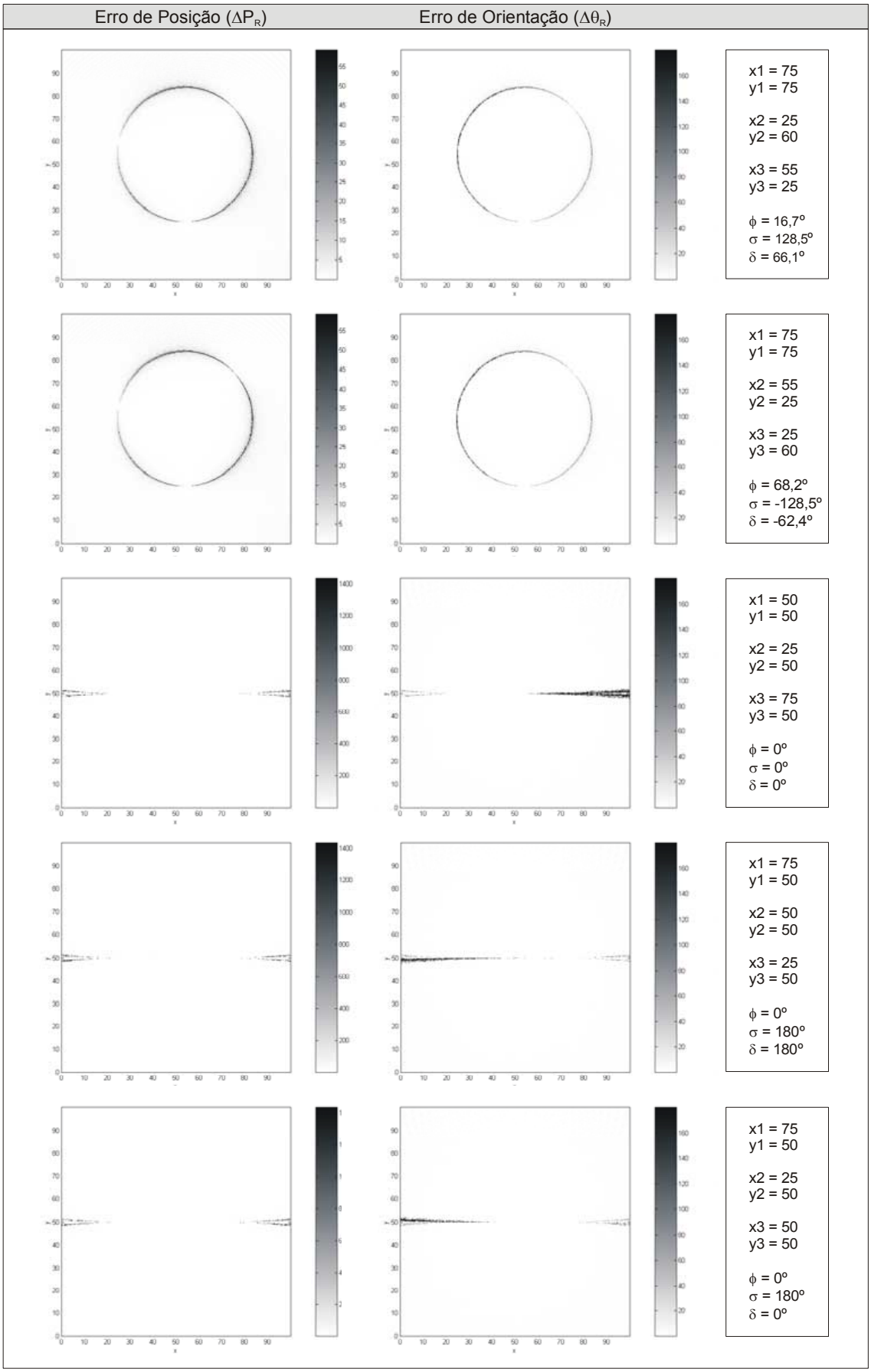


Figura 6.10: Resultados do primeiro conjunto de testes, para cada tipo de configuração de balizas.

6.3.2 Segundo Conjunto de Testes

O segundo conjunto de testes, dividido em duas séries, serve para verificar de que modo é que os erros de medição dos ângulos afectam os erros de posição e de orientação, quando um robô se encontra próximo ou afastado das balizas:

- Os três primeiros testes avaliam os erros de localização que ocorrem nas imediações de balizas colocadas no plano por forma a manter entre si uma distância próxima a metade do lado do recinto de navegação quadrado.

- Os outros três testes avaliam os erros de localização que ocorrem longe de balizas colocadas perto do centro do recinto de navegação quadrado, mantendo entre si uma distância de cerca de 1/100 desse recinto.

Em cada teste, três balizas numeradas 1, 2 e 3 são colocadas em posições conhecidas de um plano no qual está definido um referencial ortonormado x0y. As posições das balizas estão indicadas junto dos resultados de cada teste. Coloca-se um robô pontual na origem do referencial e atribui-se à sua orientação um valor aleatoriamente escolhido, superior a -180º e inferior ou igual a 180º.

De seguida, realiza-se a sequência de quatro passos representada na Figura 6.9 para posições do robô cobrindo um quadrado de 100 × 100 unidades de comprimento, com incrementos de 0,1 unidades de comprimento tanto na direcção *x* como na direcção *y*. Em cada ponto, atribui-se à orientação do robô um valor aleatoriamente escolhido, superior a -180º e inferior ou igual a 180º.

Relativamente ao primeiro conjunto de testes, a diferença está no passo II, que agora é o seguinte:

II. Os ângulos λ_1, λ_2 e λ_3 são arredondados para n casas decimais, simulando as saídas de um goniómetro digital com uma resolução ρ igual a $1º/10^n$. A correspondente incerteza de medição ($\pm\Delta\lambda$) dos ângulos é de $\pm\rho/2$.

Os primeiros testes de cada série realizam-se com n=2, os segundos com n=1 e os terceiros com n=0. Os respectivos valores de ρ e $\Delta\lambda$ encontram-se na Tabela 6.1.

Tabela 6.1: Efeitos do arredondamento de λ_1, λ_2 e λ_3.

Número de casas decimais de λ_{1m}, λ_{2m} e λ_{3m} n	**Intervalo de variação de λ_{1m}, λ_{2m} e λ_{3m}**	**Número máximo de algarismos significativos de λ_{1m}, λ_{2m} e λ_{3m}**	**Resolução do goniómetro ρ**	**Incerteza associada a λ_{1m}, λ_{2m} e λ_{3m} $\pm\Delta\lambda$**
2	[0.00º, 360.00º[	5	0,01º	±0,005º
1	[0.0º, 360.0º[	4	0,1º	±0,05º
0	[0º, 360º[	3	1º	±0,5º

Os erros de posição e de orientação obtidos em cada ponto encontram-se representados em gráficos bidimensionais e tridimensionais usando a escala multicolor indicada junto de cada gráfico (Figura 6.11, Figura 6.12, Figura 6.13 e Figura 6.14).

Os eixos dos gráficos relativos aos erros de posição estão graduados nas mesmas unidades de comprimento.

Os eixos dos *zz* dos gráficos relativos aos erros de orientação estão graduados em graus.

Para realçar os erros menores que ocorreram no quadrado analisado, foram impostos limites máximos aos valores visualizados tanto nos gráficos bidimensionais como nos gráficos tridimensionais (Tabela 6.2 e Tabela 6.3). Apenas os erros de posição e de orientação que ocorrem nas regiões próximas da circunferência definida pelas três balizas são demasiado grandes para poderem ser representados.

Tabela 6.2: Maiores valores visualizados na região próxima das balizas.

Resolução do goniómetro ρ	**Incerteza associada a λ_{1m}, λ_{2m} e λ_{3m} $\pm\Delta\lambda$**	**Maior valor visualizado (posição)**	**Maior valor visualizado (orientação)**
0,01º	±0,005º	0.05	0.06º
0,1º	±0,05º	0.5	0.6º
1º	±0,5º	5	6º

Tabela 6.3: Maiores valores visualizados na região afastada das balizas.

Resolução do goniómetro ρ	**Incerteza associada a** λ_{1m}, λ_{2m} **e** λ_{3m} $\pm\Delta\lambda$	**Maior valor visualizado (posição)**	**Maior valor visualizado (orientação)**
0,01º	±0,005º	0.2	0.2º
0,1º	±0,05º	2	2º
1º	±0,5º	20	20º

Eis algumas propriedades dos erros obtidos:

- Concordam com a análise feita previamente;
- São pequenos dentro do triângulo formado pelas três balizas;
- Aumentam significativamente à medida que o robô se aproxima da circunferência definida pelas três balizas;
- Decaem abruptamente quando o robô se afasta desta circunferência numa direcção radial e permanecem pequenos nas suas vizinhanças;
- Voltam a crescer, de forma mais suave, à medida que o robô se continua a afastar das balizas;
- Aumentam cerca de dez vezes quando a incerteza de medição de ângulos é multiplicada por dez[10].

Os resultados sugerem que, se a incerteza de medição de ângulos for suficientemente pequena, o robô será capaz de se localizar, cometendo apenas pequenos erros de localização, numa extensa área do plano de navegação.

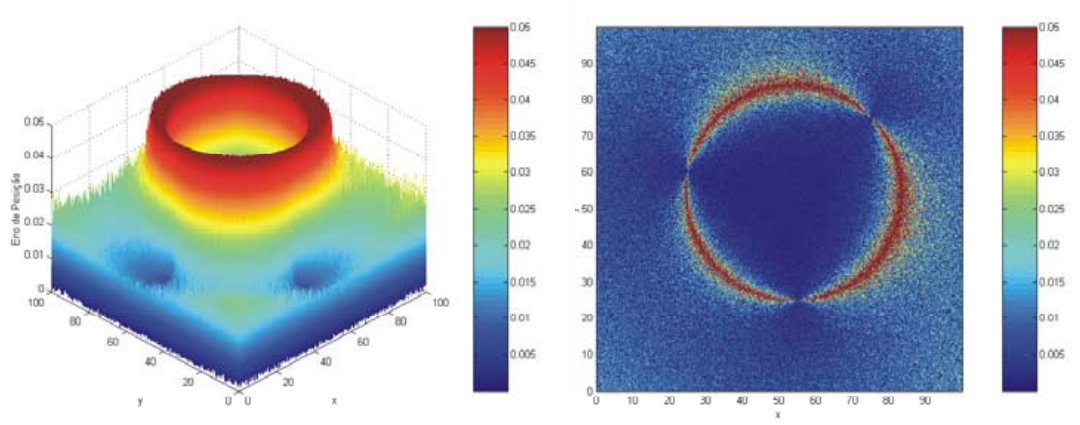
[10] A validade desta regra foi constatada em testes (realizados integralmente em dupla precisão) com ρ a variar entre 0,00001º e 1º. Os resultados obtidos com ρ inferior a 0.01º não são mostrados.

Erro de Posição (ΔP_R)

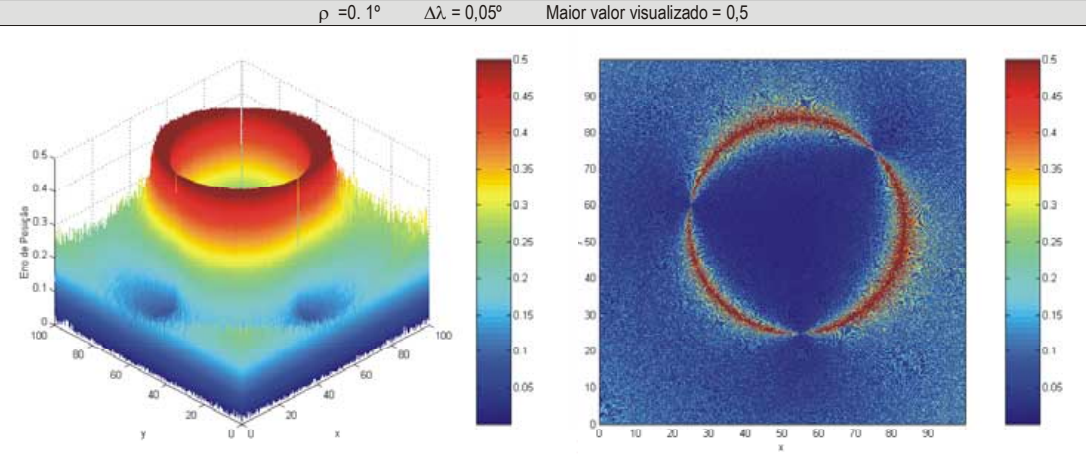
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 75 y1 = 75
x2 = 25 y2 = 60
x3 = 55 y3 = 25

Distância entre as balizas 1 e 2 = 52,2
Distância entre as balizas 1 e 3 = 53,9
ϕ = 16,7º
σ = 128,5º
δ = 66,1º

ρ = 0,01º $\Delta\lambda$ = 0,005º Maior valor visualizado = 0,05



ρ =0. 1º $\Delta\lambda$ = 0,05º Maior valor visualizado = 0,5



ρ = 1º $\Delta\lambda$ = 0,5º Maior valor visualizado = 5

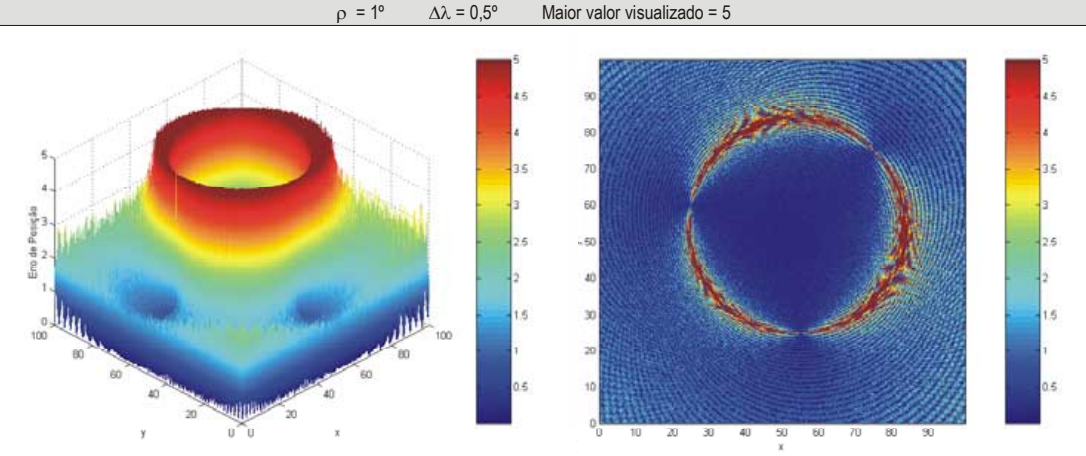


Figura 6.11: Erros de posição na região próxima das balizas.

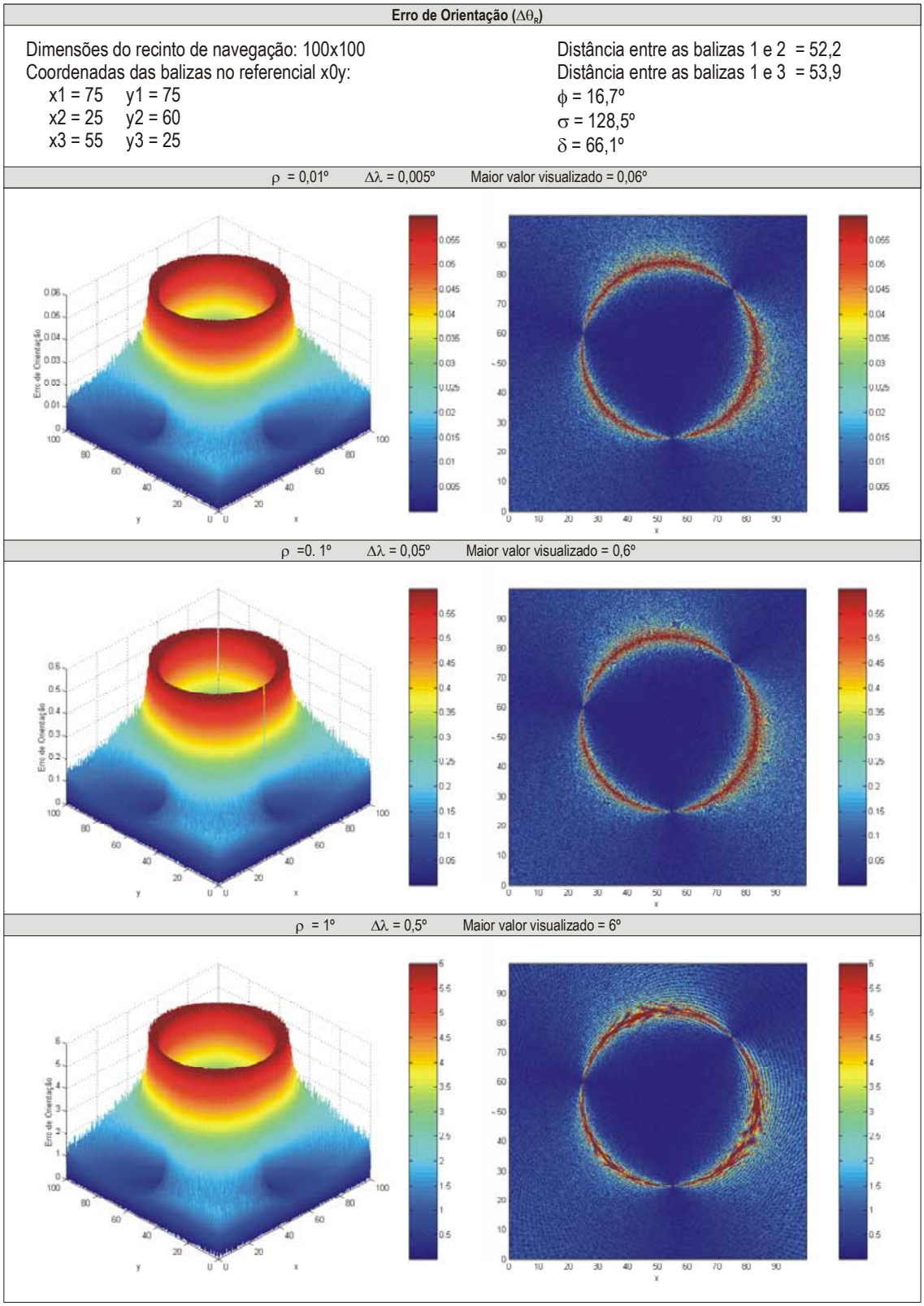


Figura 6.12: Erros de orientação na região próxima das balizas.

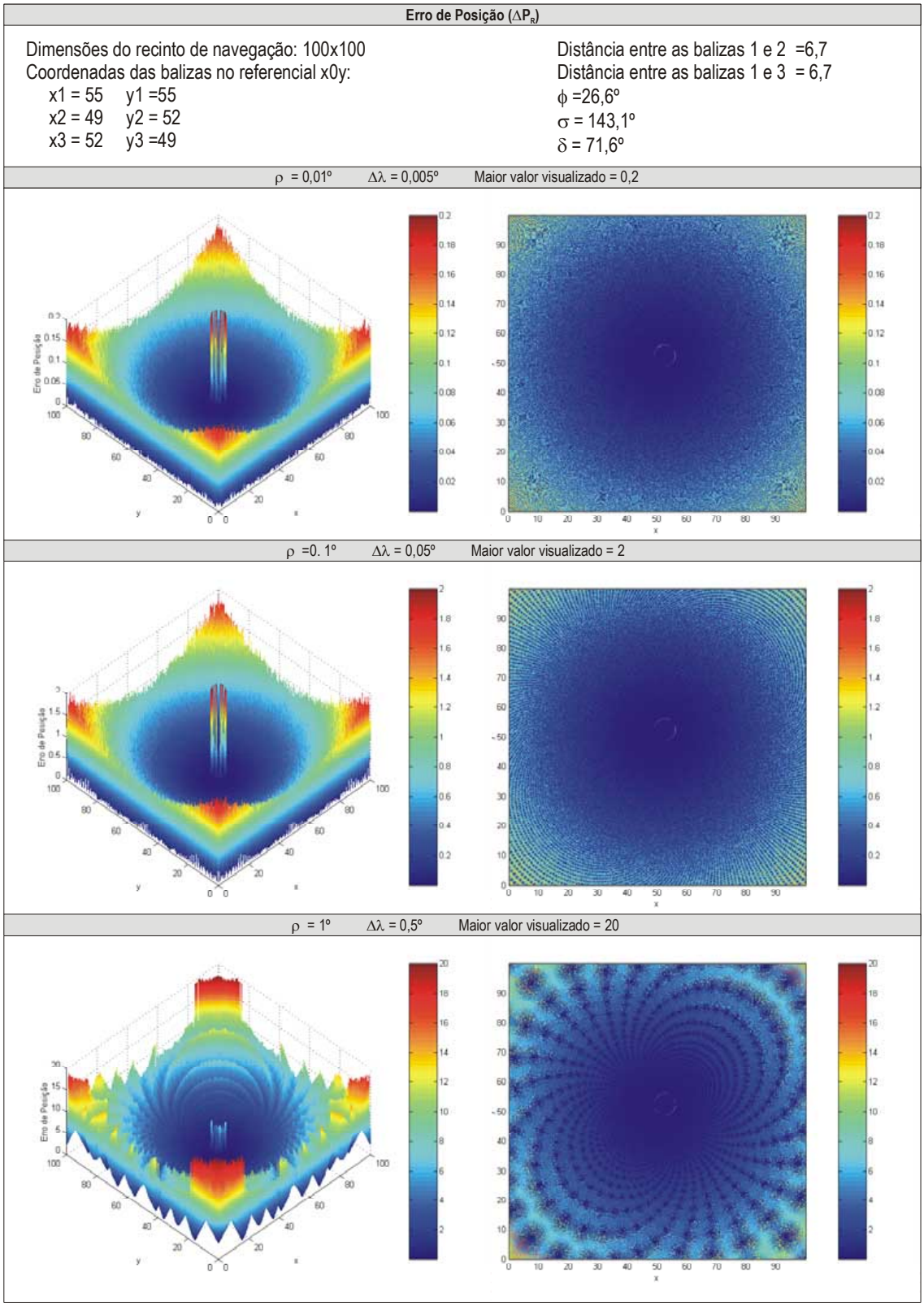


Figura 6.13: Erros de posição na região afastada das balizas.

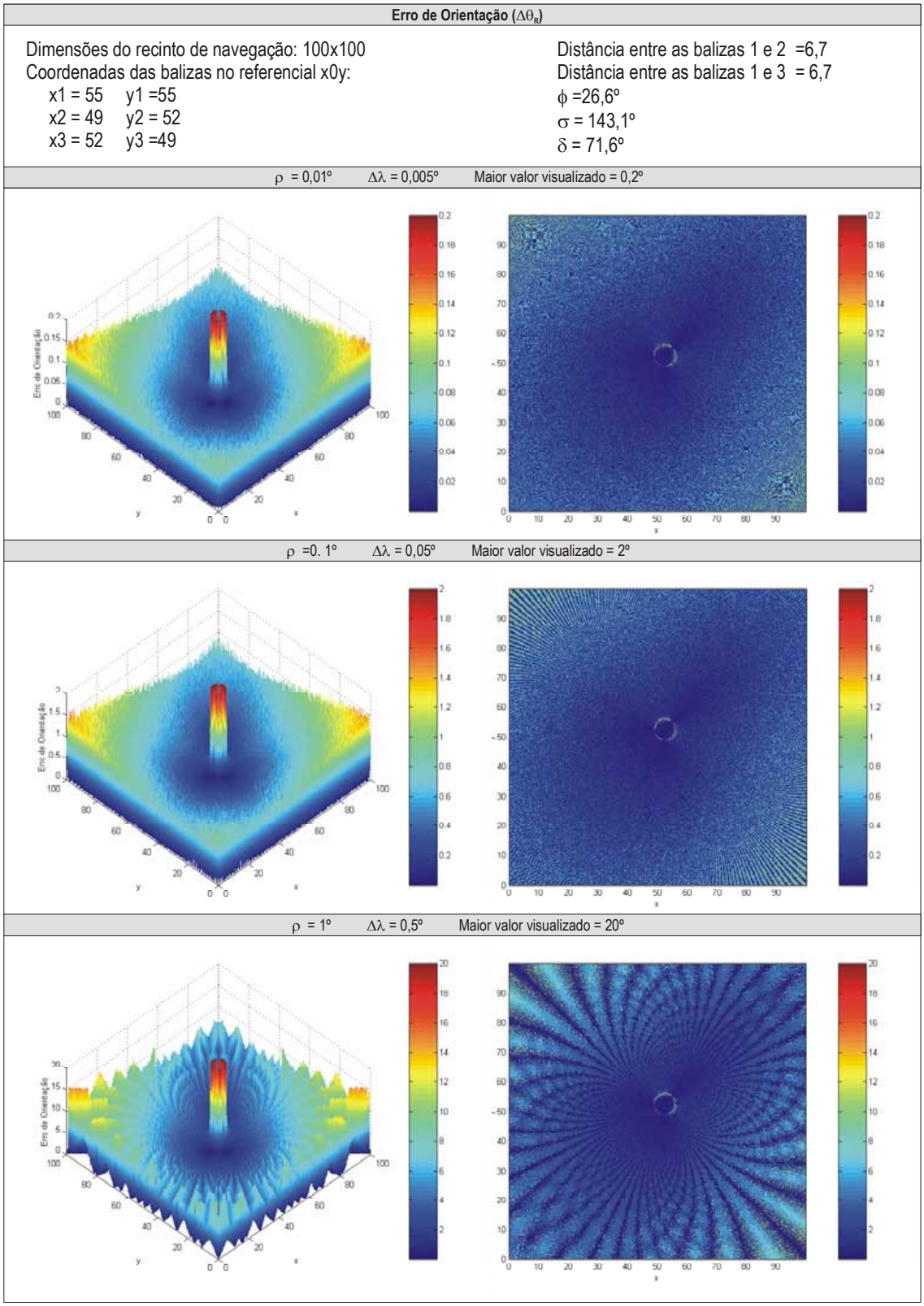


Figura 6.14: Erros de orientação na região afastada das balizas.

Um cuidado importante a ter ao executar os programas de teste é garantir que os cálculos são efectuados com um número suficiente de algarismos significativos, para evitar a degradação dos resultados devida a erros nos cálculos. Com isto presente, fez-se uma verificação recorrendo aos dois formatos em vírgula flutuante IEEE 754 (Kahan, 1996; NCG, 1996), disponíveis em *Java 2* (Tabela 6.4):

- Em primeiro lugar realizaram-se testes usando dupla precisão (53 bits significativos) em todos os quatro passos;

- Depois repetiram-se os mesmos testes mas, desta vez, usando precisão simples (24 bits significativos) no passo III.

Para ρ a variar entre 0.01º e 1º não foi possível distinguir os resultados obtidos em cada um dos dois conjuntos de testes. Isto mostra que os erros de localização se devem essencialmente aos erros produzidos no passo II e não a erros nos cálculos.

Os gráficos apresentados resultam de testes realizados usando sempre dupla precisão.

Tabela 6.4: Dígitos significativos dos formatos em vírgula flutuante IEEE 754, disponíveis em *Java 2*.

Formato em vírgula flutuante	Bits significativos	Algarismos decimais significativos
IEEE 754 Single	24	6 – 9
IEEE 754 Double	53	15 – 17

6.3.3 Terceiro Conjunto de Testes

O terceiro conjunto de testes, num total de seis, destina-se a validar o método de caracterização das incertezas de posição e de orientação que foi proposto no Capítulo 5.

Em cada teste, três balizas numeradas 1, 2 e 3 são colocadas em posições conhecidas de um plano no qual está definido um referencial ortonormado x0y. As posições das balizas estão indicadas junto dos resultados de cada teste. Coloca-se um robô pontual na origem do referencial e atribui-se à sua orientação um valor aleatoriamente escolhido, superior a -180º e inferior ou igual a 180º. De seguida, realiza-se a sequência de seis passos representada na Figura 6.15:

I. Os ângulos λ_1, λ_2 e λ_3 são calculados a partir das posições – conhecidas à partida – das balizas e do robô.

II. Os ângulos λ_1, λ_2 e λ_3 são arredondados para números inteiros, simulando as saídas de um goniómetro digital com uma resolução ρ igual a 1º. A correspondente incerteza de medição dos ângulos ($\pm\Delta\lambda$) é de ±0,5º.

III. O Algoritmo Generalizado de Triangulação Geométrica calcula a posição e a orientação do robô a partir dos valores arredondados de λ_1, λ_2 e λ_3 e das posições – conhecidas *a priori* – das balizas.

IV. O erro de posição ΔP_R e o erro de orientação $\Delta\theta_R$ são calculados comparando a posição e a orientação do robô conhecidas *a priori* com a posição e a orientação do robô determinadas pelo Algoritmo Generalizado de Triangulação Geométrica.

V. O erro máximo de posição $\Delta P_{Rmáx}$ e o erro máximo de orientação $\Delta\theta_{Rmáx}$ são calculados utilizando o método descrito no Capítulo 5.

VI. Calcula-se a diferença entre o erro de posição e o erro máximo de posição ($\Delta P_R - \Delta P_{Rmáx}$), e a diferença entre o erro de orientação e o erro máximo de orientação ($\Delta\theta_R - \Delta\theta_{Rmáx}$).

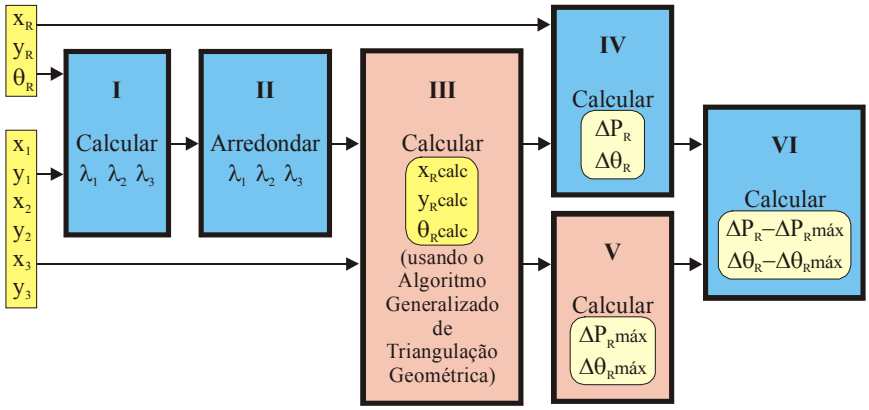


Figura 6.15: Sequência de passos executados no terceiro conjunto de testes.

Repetem-se os seis passos para posições do robô cobrindo um quadrado de 100 × 100 unidades de comprimento, com incrementos de 0,1 unidades de comprimento tanto na direcção *x* como na direcção *y*. Em cada ponto, atribui-se à orientação do robô um valor aleatoriamente escolhido, superior a -180º e inferior ou igual a 180º.

Sejam $\Delta P_R(x,y)$, $\Delta\theta_R(x,y)$, $\Delta P_{Rmáx}(x,y)$ e $\Delta\theta_{Rmáx}(x,y)$ quatro funções de x e y cujos valores em cada ponto do plano de navegação são, respectivamente, ΔP_R, $\Delta\theta_R$, $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$, calculados de acordo com a sequência de passos da Figura 6.15.

As funções $\Delta P_R(x,y)$, $\Delta\theta_R(x,y)$, $\Delta P_{Rmáx}(x,y)$, $\Delta\theta_{Rmáx}(x,y)$, $\Delta P_R(x,y)$-$\Delta P_{Rmáx}(x,y)$, e $\Delta\theta_R(x,y)$-$\Delta\theta_{Rmáx}(x,y)$ encontram-se representadas em gráficos bidimensionais e tridimensionais usando a escala multicolor indicada junto de cada gráfico (Figura 6.16 e seguintes, até à Figura 6.27). As regiões representadas a branco são constituídas por pontos do plano de navegação nos quais não é possível calcular $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$ recorrendo ao método proposto. A presença do robô numa dessas regiões – que incluem a circunferência definida por três balizas não colineares e a recta definida por três balizas colineares – é detectada pelo algoritmo, pelo que não há erros de localização. Simplesmente, os cálculos não são realizados e, como aviso, o algoritmo produz um resultado NaN (*Not-a-Number*).

O método de cálculo de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$ funcionou sempre correctamente em inúmeros testes realizados. Na Figura 6.16 e seguintes, até à Figura 6.27, apresentam-se apenas os resultados de testes correspondentes às seguintes situações:

- Autolocalização na região próxima de três balizas não colineares ordenadas no sentido directo (Figura 6.16 e Figura 6.17);
- Autolocalização na região próxima de três balizas não colineares ordenadas no sentido inverso (Figura 6.18 e Figura 6.19);
- Autolocalização na região afastada de três balizas não colineares ordenadas no sentido directo (Figura 6.20 e Figura 6.21);
- Autolocalização na região próxima de três balizas colineares, estando a baliza 1 entre as balizas 2 e 3 (Figura 6.22 e Figura 6.23);
- Autolocalização na região próxima de três balizas colineares, estando a baliza 2 entre as balizas 1 e 3 (Figura 6.24 e Figura 6.25);
- Autolocalização na região próxima de três balizas colineares, estando a baliza 3 entre as balizas 1 e 2 (Figura 6.26 e Figura 6.27).

Os eixos dos gráficos relativos a $\Delta P_R(x,y)$, $\Delta P_{Rmáx}(x,y)$, e $\Delta P_R(x,y)$-$\Delta P_{Rmáx}(x,y)$ estão graduados nas mesmas unidades de comprimento.

Os eixos dos *xx* e dos *yy* dos gráficos relativos a $\Delta\theta_R(x,y)$, $\Delta\theta_{Rmáx}(x,y)$ e $\Delta\theta_R(x,y)$-$\Delta\theta_{Rmáx}(x,y)$ estão graduados nas mesmas unidades de comprimento. Os eixos dos *zz* desses gráficos estão graduados em graus.

Em cada gráfico impuseram-se os limites máximos aos valores visualizados referidos na Tabela 6.2 e na Tabela 6.3 (linhas correspondentes a $\rho = 1º$).

Se ΔP_R ultrapassar $\Delta P_{Rmáx}$ em algum dos pontos testados, então o algoritmo faz com que, no respectivo ponto do gráfico de $\Delta P_R(x,y)$-$\Delta P_{Rmáx}(x,y)$, seja representado o limite máximo dos valores visualizados. De igual forma, se $\Delta\theta_R$ ultrapassar $\Delta\theta_{Rmáx}$ em algum dos pontos testados, então o algoritmo faz com que, no respectivo ponto do gráfico de $\Delta\theta_R(x,y)$-$\Delta\theta_{Rmáx}(x,y)$, seja representado o limite máximo dos valores visualizados. Facilmente se constata que, nos testes cujos resultados se apresentam, nunca ocorre nenhuma destas situações. Foram realizados muitos outros testes, com diferentes configurações de balizas e vários valores de ρ. Não houve uma única vez em que ΔP_R tenha sido superior a $\Delta P_{Rmáx}$ ou $\Delta\theta_R$ tenha sido superior a $\Delta\theta_{Rmáx}$.

Em geral, a função $\Delta P_{Rmáx}(x,y)$ constitui uma boa envolvente da função $\Delta P_R(x,y)$, excepto nas imediações da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares. Aí verifica-se que os valores de $\Delta P_{Rmáx}(x,y)$ calculados em cada ponto ultrapassam largamente os respectivos valores de $\Delta P_R(x,y)$.

A função $\Delta\theta_{Rmáx}(x,y)$ só é tão boa envolvente da função $\Delta\theta_R(x,y)$ em algumas zonas do plano de navegação ou então se forem utilizadas configurações particulares de balizas (por exemplo, a que corresponde à Figura 6.22 e à Figura 6.23). Isto evidencia a pertinência de se estudar a relação entre τ e λ_1 tendo em vista tornar a função $\Delta\theta_{Rmáx}(x,y)$ uma melhor envolvente da função $\Delta\theta_R(x,y)$. Os valores de $\Delta\theta_{Rmáx}(x,y)$ calculados nas imediações da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares também ultrapassam largamente os respectivos valores de $\Delta\theta_R(x,y)$.

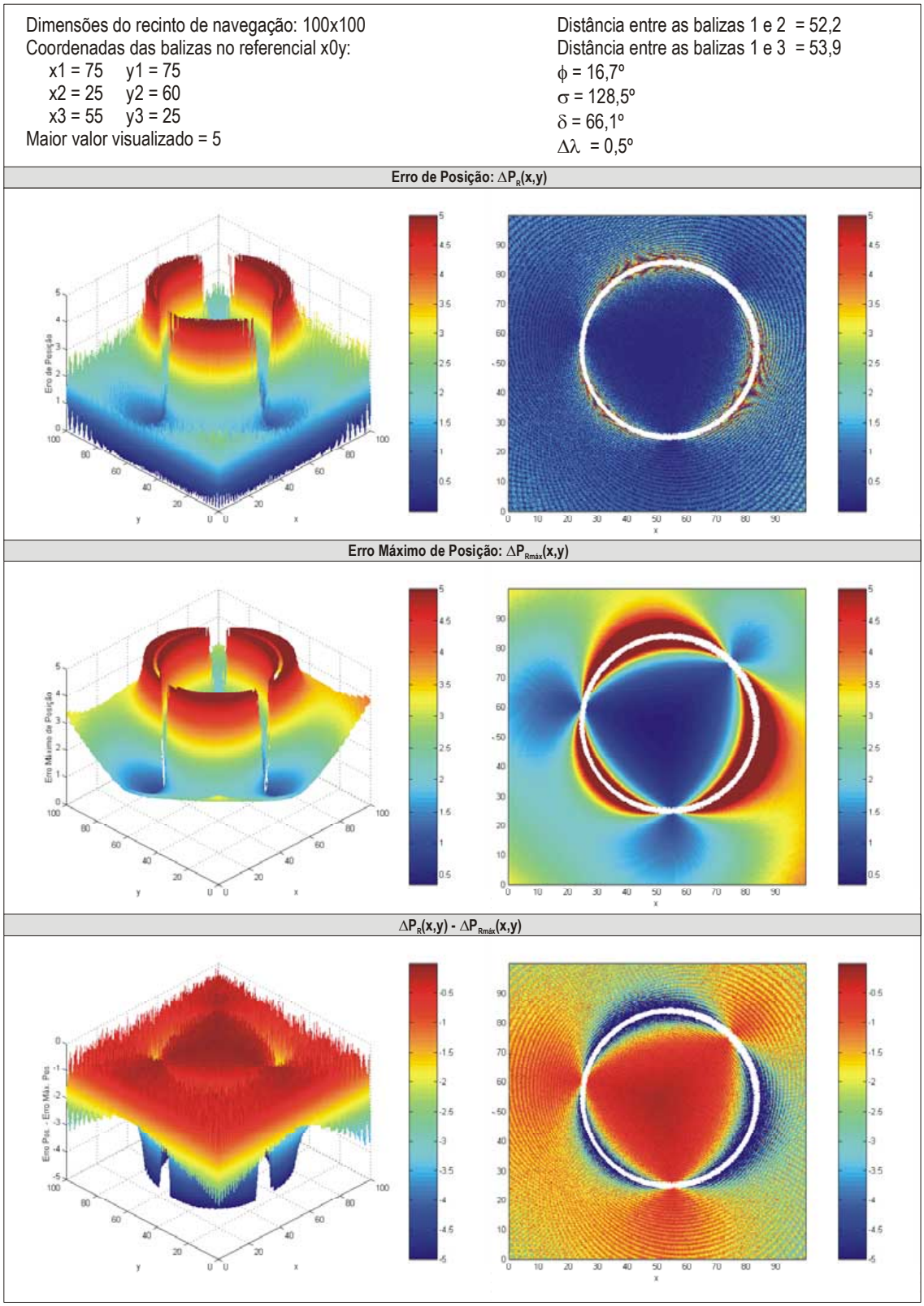
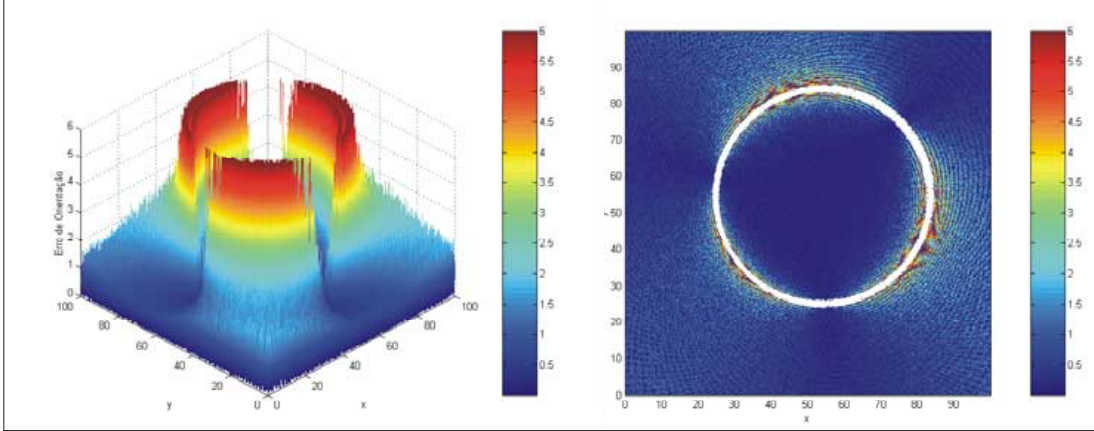


Figura 6.16: Erro de posição, erro máximo de posição e diferença entre o erro de posição e o erro máximo de posição, na região próxima de três balizas não colineares ordenadas no sentido directo.

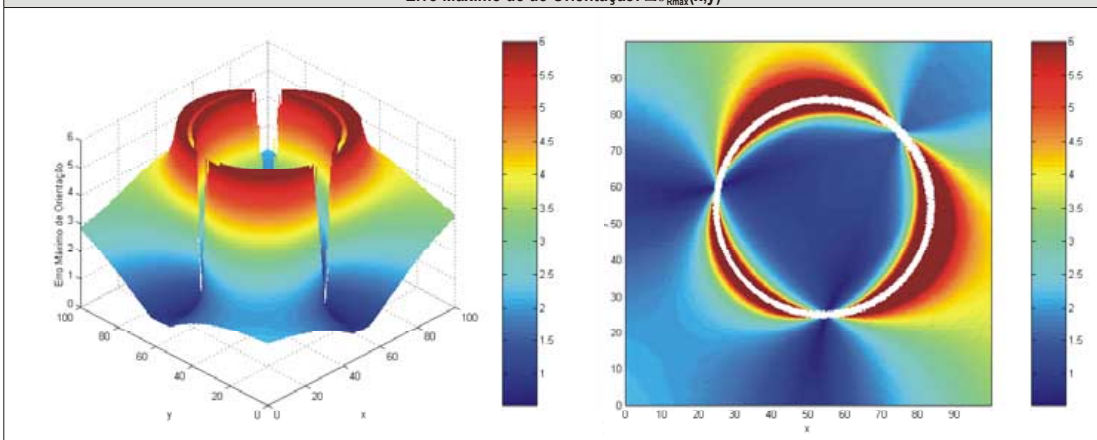
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 75 y1 = 75
x2 = 25 y2 = 60
x3 = 55 y3 = 25
Maior valor visualizado = 6º

Distância entre as balizas 1 e 2 = 52,2
Distância entre as balizas 1 e 3 = 53,9
ϕ = 16,7º
σ = 128,5º
δ = 66,1º
$\Delta\lambda$ = 0,5º

Erro de Orientação: $\Delta\theta_R(x,y)$



Erro Máximo de de Orientação: $\Delta\theta_{Rmáx}(x,y)$



$\Delta\theta_R(x,y)$ - $\Delta\theta_{Rmáx}(x,y)$

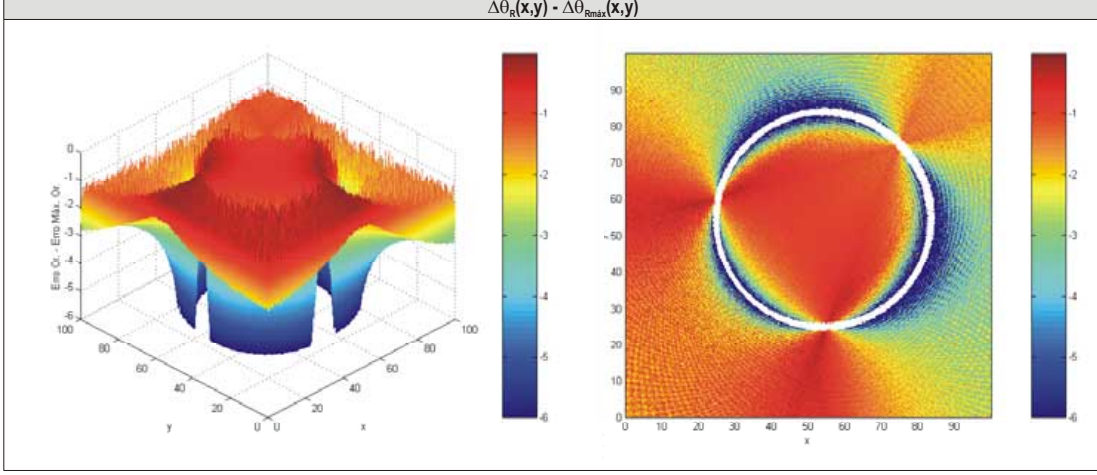
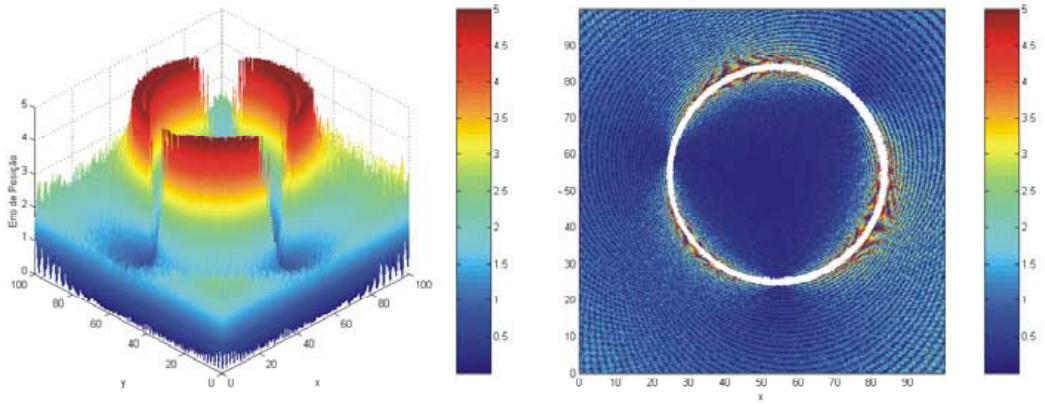


Figura 6.17: Erro de orientação, erro máximo de orientação e diferença entre o erro de orientação e o erro máximo de orientação, na região próxima de três balizas não colineares ordenadas no sentido directo.

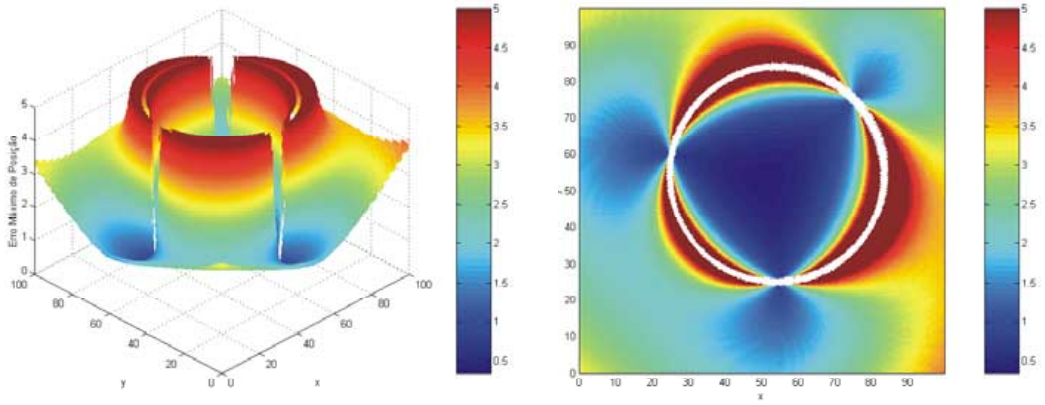
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 75 y1 = 75
x2 = 55 y2 = 25
x3 = 25 y3 = 60
Maior valor visualizado = 5

Distância entre as balizas 1 e 2 = 53,9
Distância entre as balizas 1 e 3 = 52,2
ϕ = 68,2º
σ = -128,5º
δ = -62,4º
$\Delta\lambda$ = 0,5º

Erro de Posição: $\Delta P_R(x,y)$



Erro Máximo de Posição: $\Delta P_{Rmáx}(x,y)$



$\Delta P_R(x,y)$ - $\Delta P_{Rmáx}(x,y)$

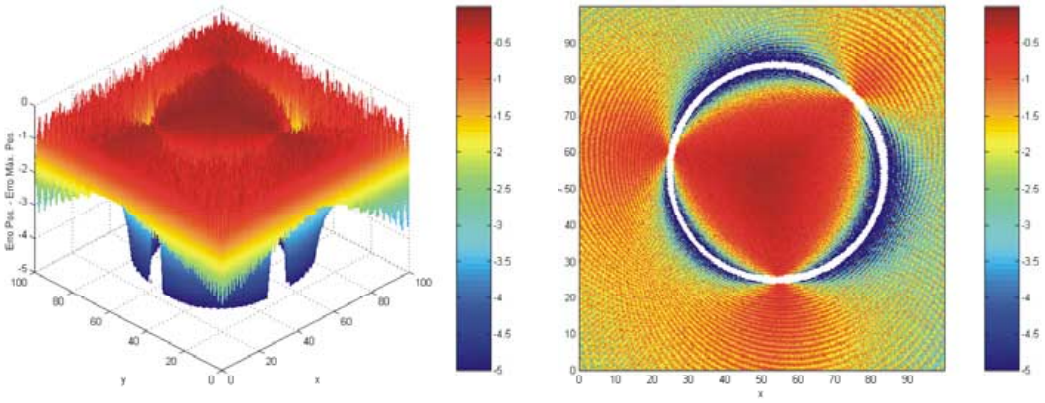
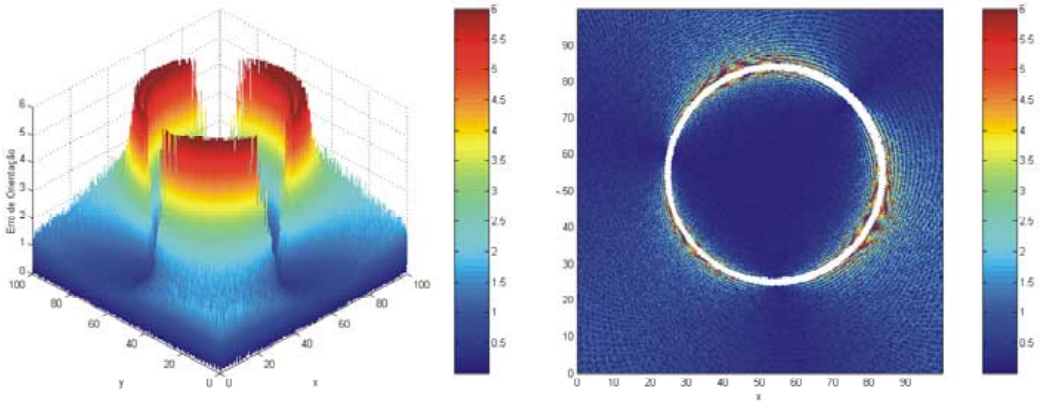


Figura 6.18: Erro de posição, erro máximo de posição e diferença entre o erro de posição e o erro máximo de posição, na região próxima de três balizas não colineares ordenadas no sentido inverso.

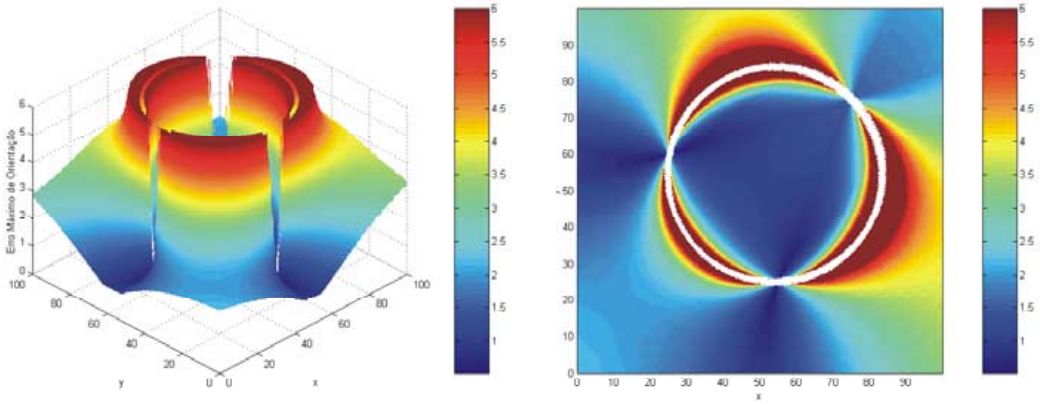
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 75 y1 = 75
x2 = 55 y2 = 25
x3 = 25 y3 = 60
Maior valor visualizado = 6º

Distância entre as balizas 1 e 2 = 53,9
Distância entre as balizas 1 e 3 = 52,2
ϕ = 68,2º
σ = -128,5º
δ = -62,4º
$\Delta\lambda$ = 0,5º

Erro de Orientação: $\Delta\theta_R(x,y)$



Erro Máximo de de Orientação: $\Delta\theta_{Rmáx}(x,y)$



$\Delta\theta_R(x,y)$ - $\Delta\theta_{Rmáx}(x,y)$

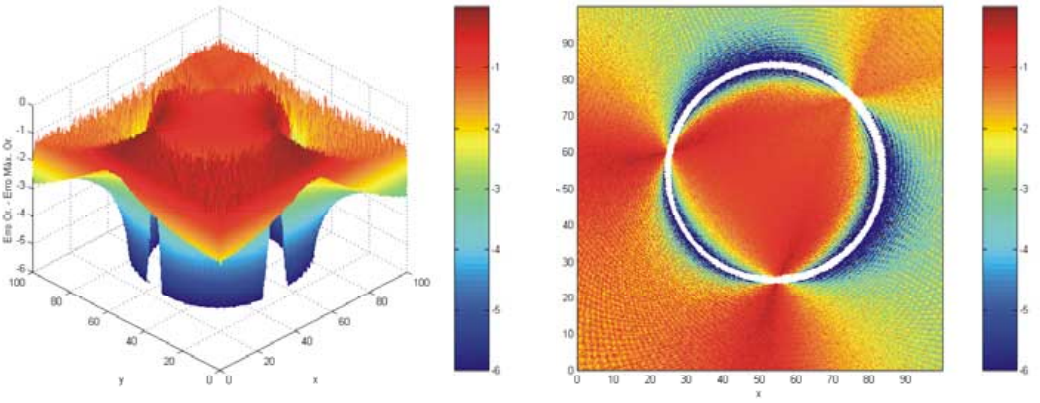
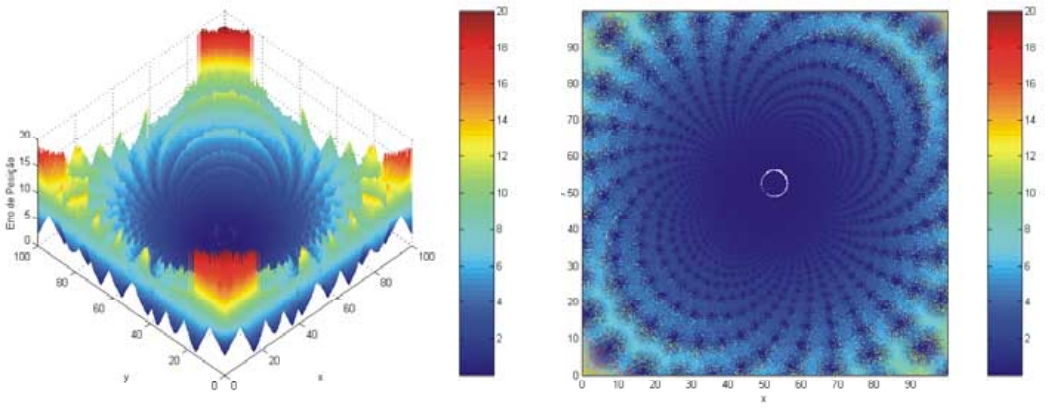


Figura 6.19: Erro de orientação, erro máximo de orientação e diferença entre o erro de orientação e o erro máximo de orientação, na região próxima de três balizas não colineares ordenadas no sentido inverso.

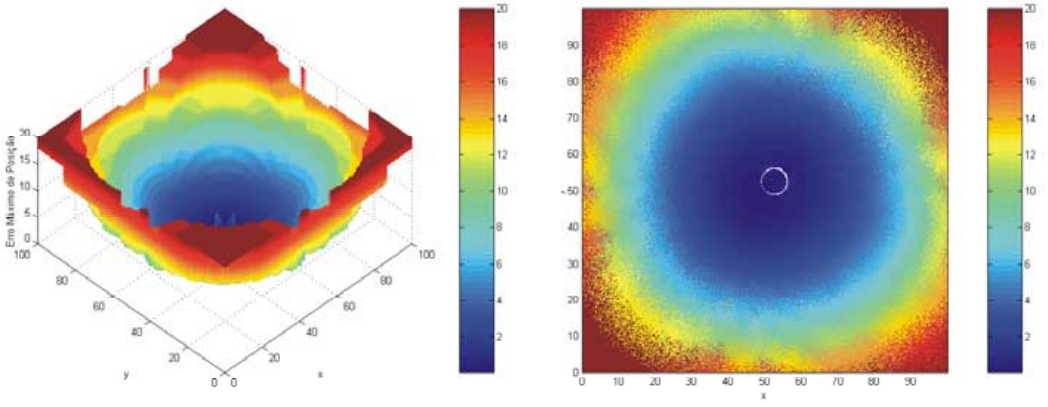
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 55 y1 =55
x2 = 49 y2 = 52
x3 = 52 y3 =49
Maior valor visualizado = 20

Distância entre as balizas 1 e 2 =6,7
Distância entre as balizas 1 e 3 = 6,7
ϕ =26,6º
σ = 143,1º
δ = 71,6º
$\Delta\lambda$ = 0,5º

Erro de Posição: $\Delta P_R(x,y)$



Erro Máximo de Posição: $\Delta P_{Rmáx}(x,y)$



$\Delta P_R(x,y)$ - $\Delta P_{Rmáx}(x,y)$

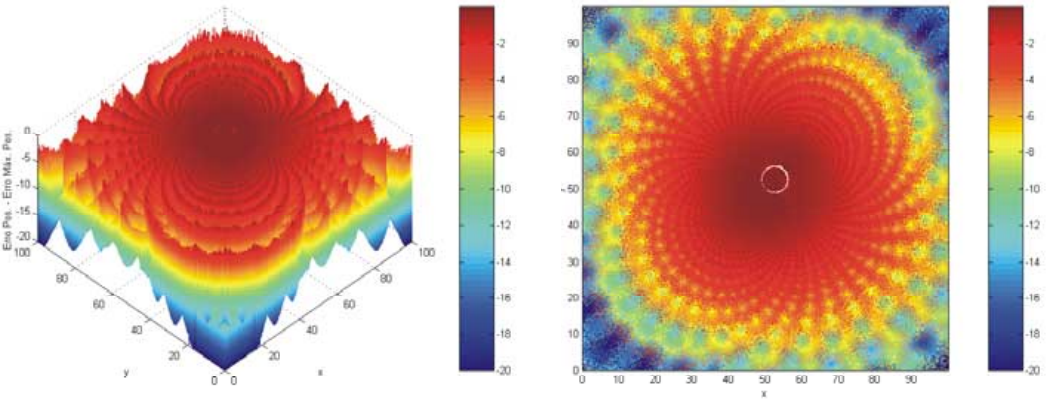


Figura 6.20: Erro de posição, erro máximo de posição e diferença entre o erro de posição e o erro máximo de posição, na região afastada de três balizas não colineares ordenadas no sentido directo.

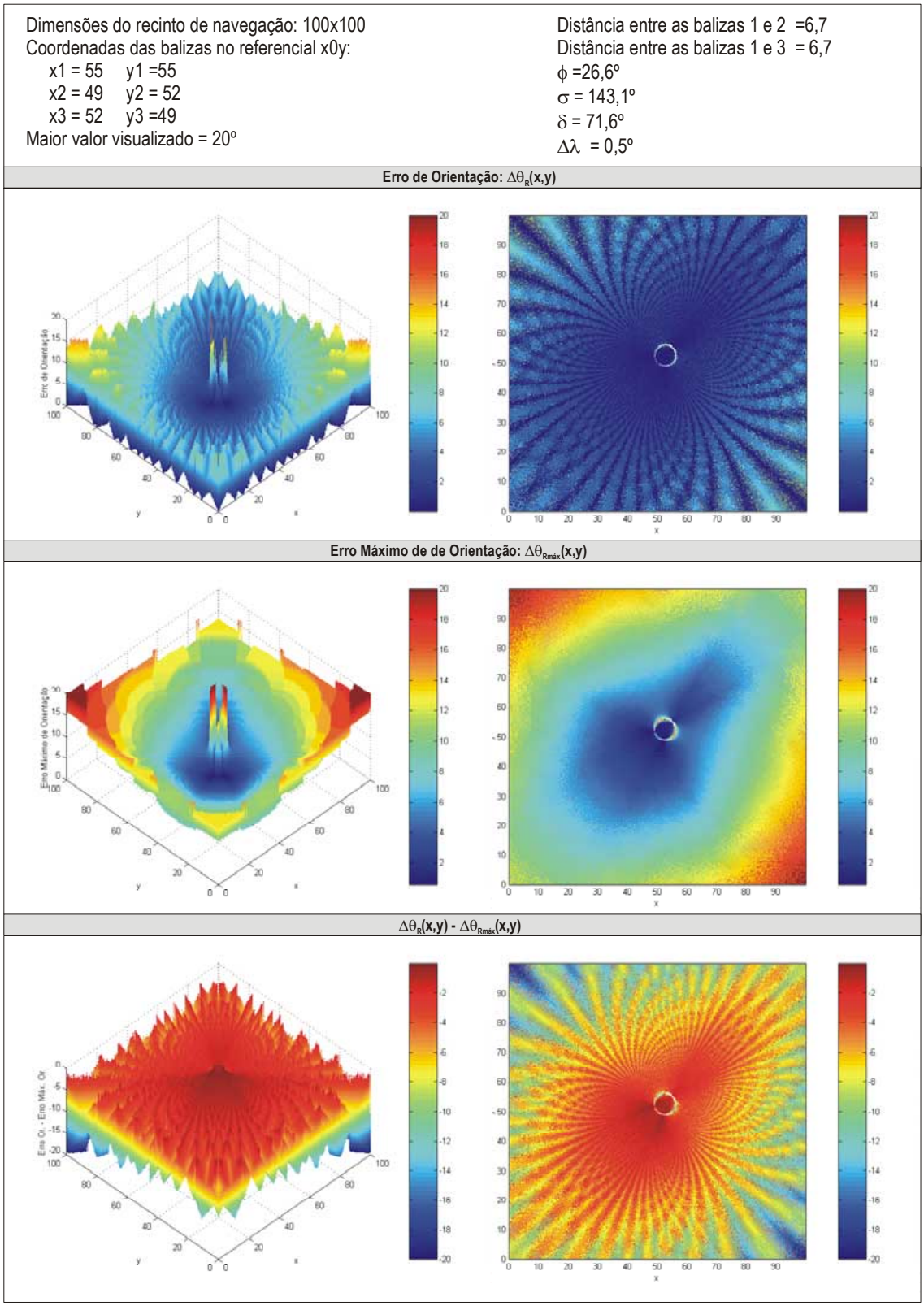
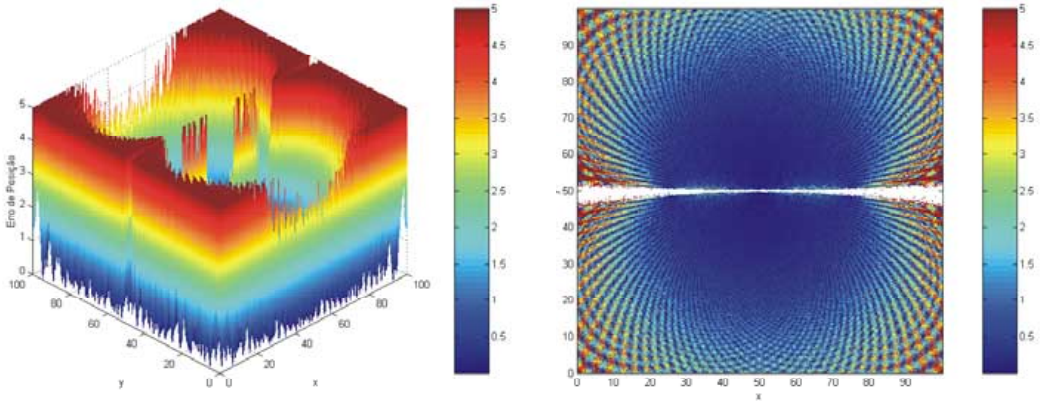


Figura 6.21: Erro de orientação, erro máximo de orientação e diferença entre o erro de orientação e o erro máximo de orientação, na região afastada de três balizas não colineares ordenadas no sentido directo.

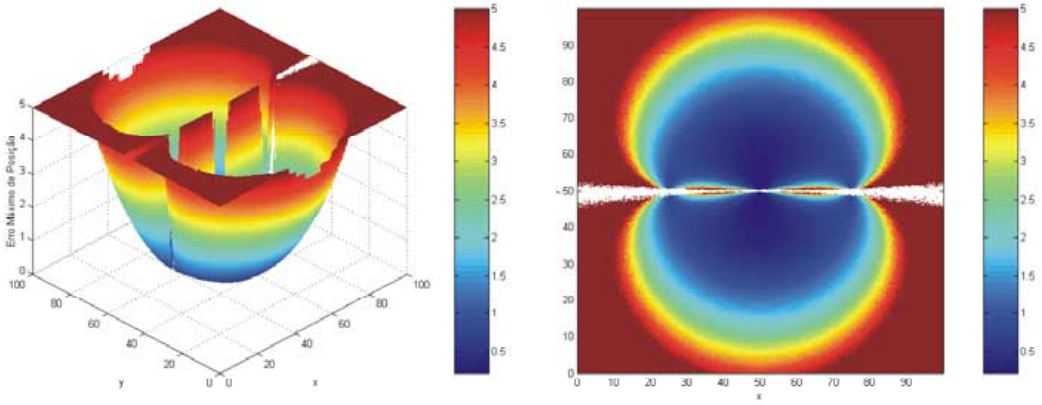
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 50 y1 = 50
x2 = 25 y2 = 50
x3 = 75 y3 = 50
Maior valor visualizado = 5

Distância entre as balizas 1 e 2 = 25
Distância entre as balizas 1 e 3 = 25
$\phi = 0º$
$\sigma = 0º$
$\delta = 0º$
$\Delta\lambda = 0{,}5º$

Erro de Posição: $\Delta P_R(x,y)$



Erro Máximo de Posição: $\Delta P_{Rmáx}(x,y)$



$\Delta P_R(x,y) - \Delta P_{Rmáx}(x,y)$

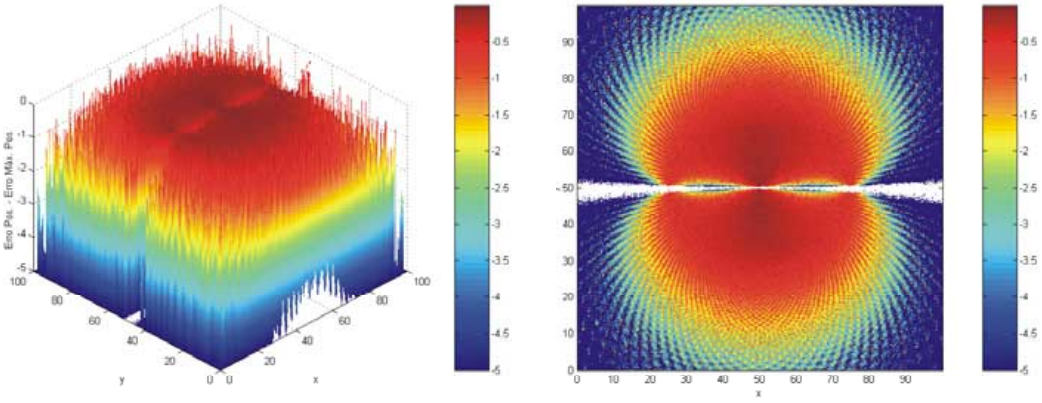
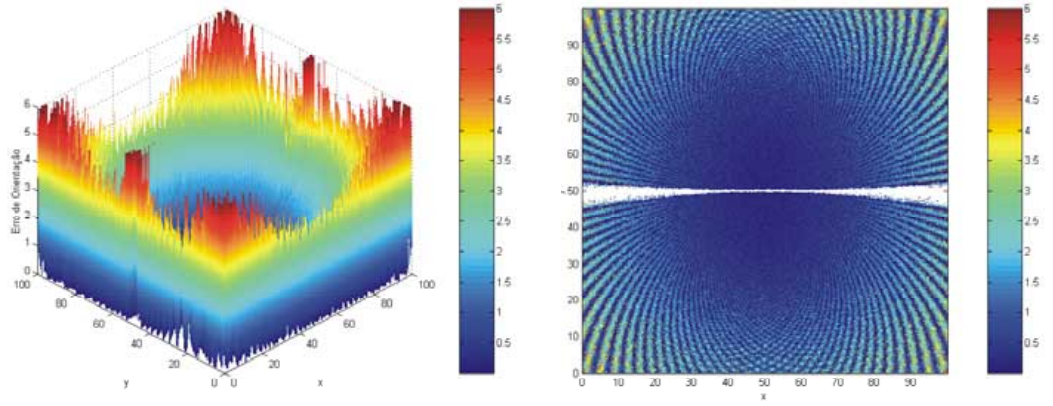


Figura 6.22: Erro de posição, erro máximo de posição e diferença entre o erro de posição e o erro máximo de posição, na região próxima de três balizas colineares, estando a baliza 1 entre as balizas 2 e 3.

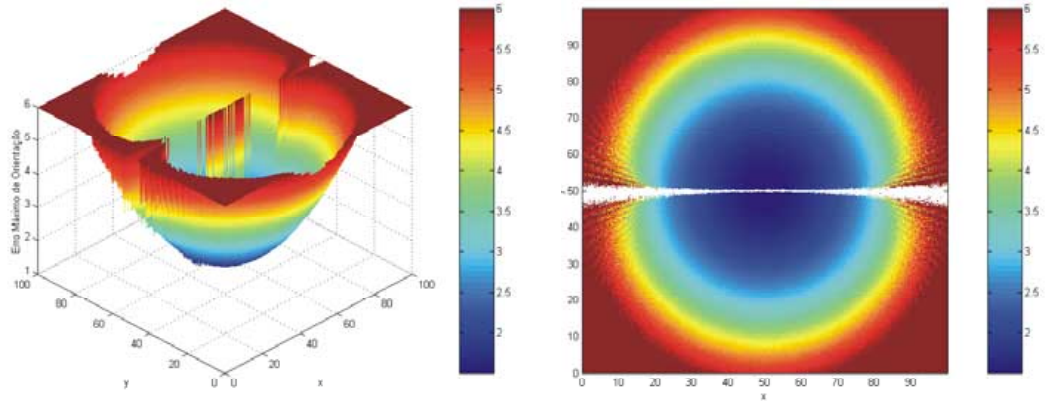
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 50 y1 = 50
x2 = 25 y2 = 50
x3 = 75 y3 = 50
Maior valor visualizado = 6º

Distância entre as balizas 1 e 2 = 25
Distância entre as balizas 1 e 3 = 25
ϕ = 0º
σ = 0º
δ = 0º
$\Delta\lambda$ = 0,5º

Erro de Orientação: $\Delta\theta_R(x,y)$



Erro Máximo de de Orientação: $\Delta\theta_{Rmáx}(x,y)$



$\Delta\theta_R(x,y)$ - $\Delta\theta_{Rmáx}(x,y)$

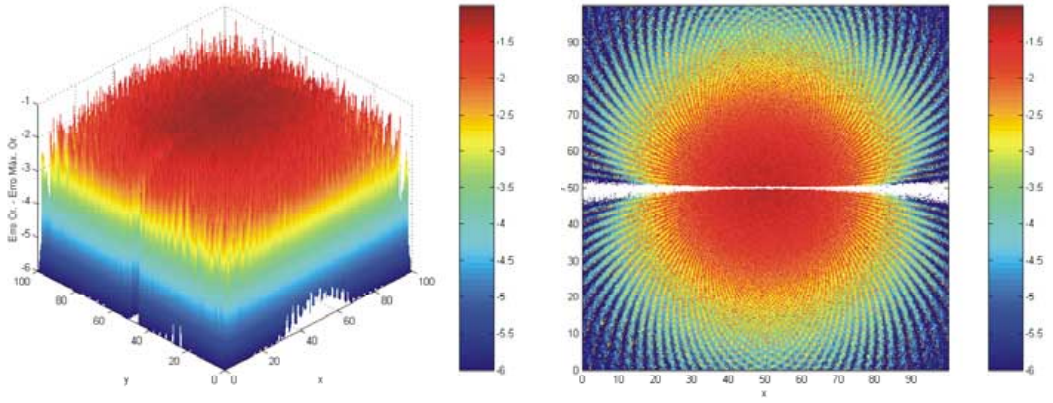


Figura 6.23: Erro de orientação, erro máximo de orientação e diferença entre o erro de orientação e o erro máximo de orientação, na região próxima de três balizas colineares, estando a baliza 1 entre as balizas 2 e 3.

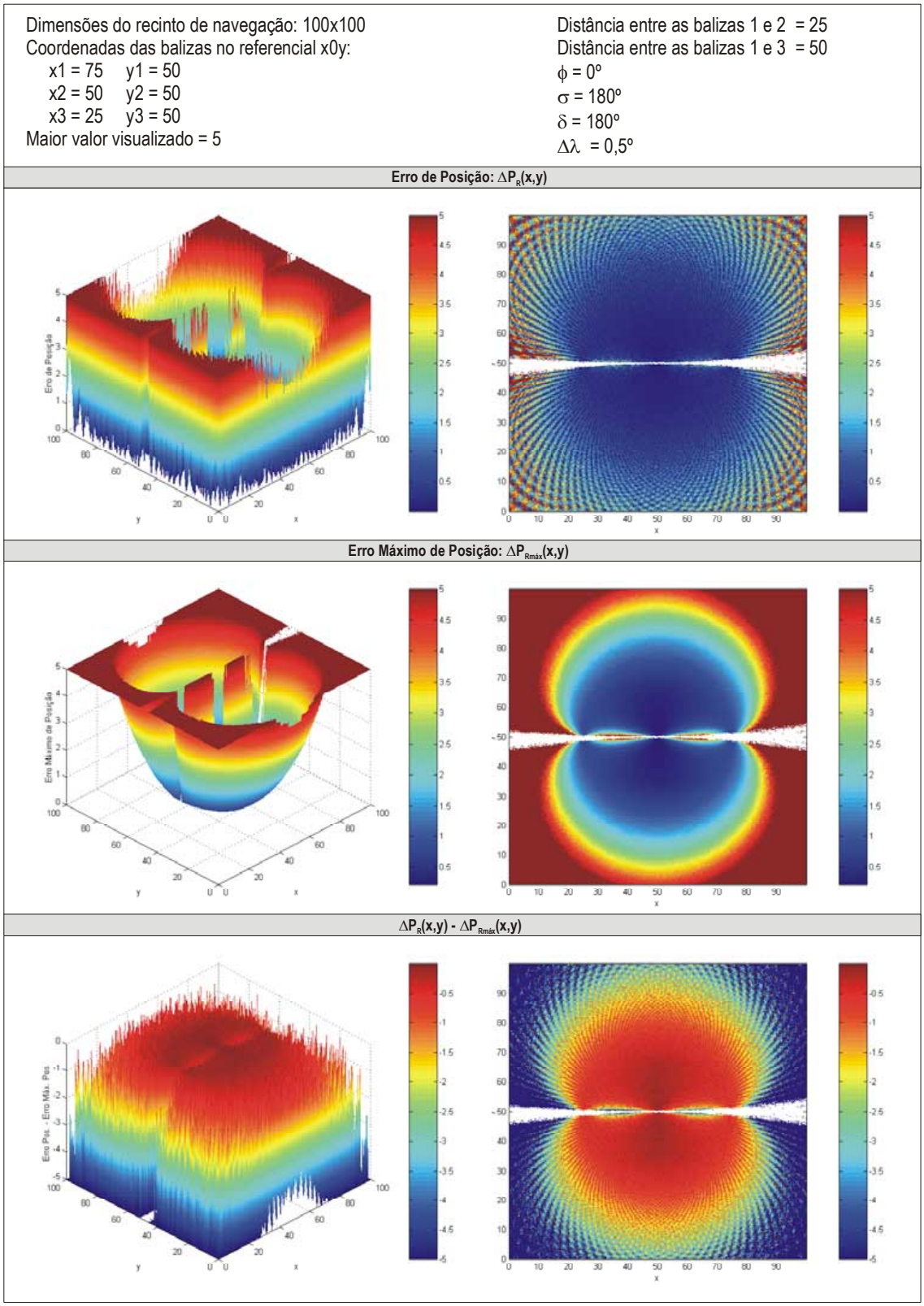
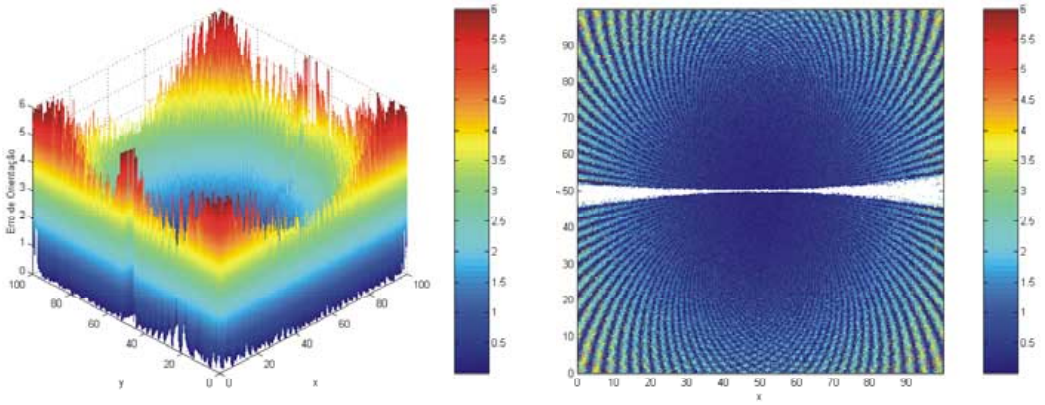


Figura 6.24: Erro de posição, erro máximo de posição e diferença entre o erro de posição e o erro máximo de posição, na região próxima de três balizas colineares, estando a baliza 2 entre as balizas 1 e 3.

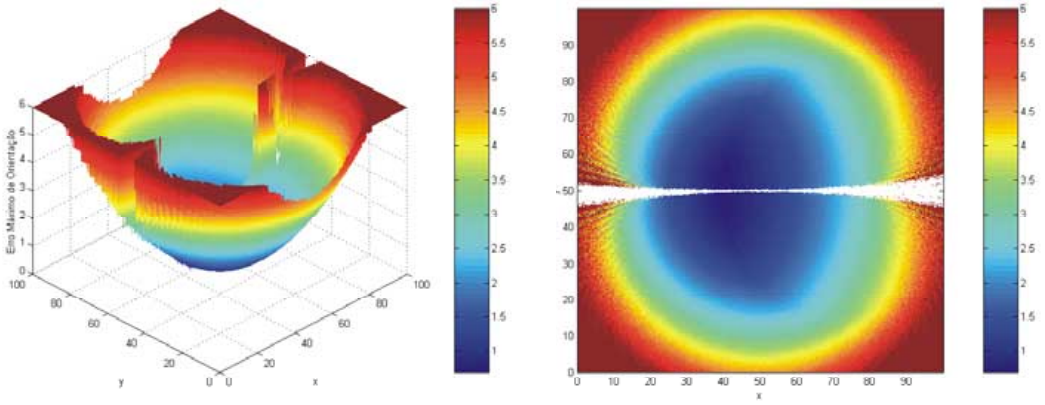
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 75 y1 = 50
x2 = 50 y2 = 50
x3 = 25 y3 = 50
Maior valor visualizado = 6º

Distância entre as balizas 1 e 2 = 25
Distância entre as balizas 1 e 3 = 50
ϕ = 0º
σ = 180º
δ = 180º
$\Delta\lambda$ = 0,5º

Erro de Orientação: $\Delta\theta_R(x,y)$



Erro Máximo de de Orientação: $\Delta\theta_{Rmáx}(x,y)$



$\Delta\theta_R(x,y)$ - $\Delta\theta_{Rmáx}(x,y)$

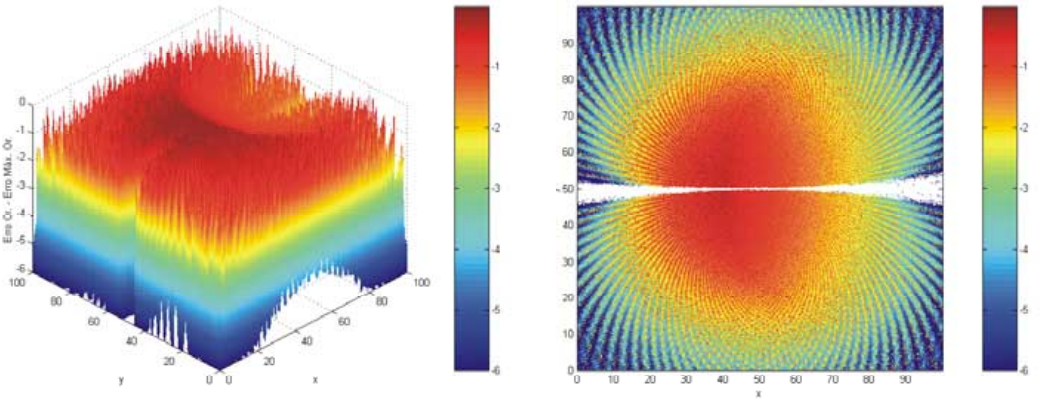


Figura 6.25: Erro de orientação, erro máximo de orientação e diferença entre o erro de orientação e o erro máximo de orientação, na região próxima de três balizas colineares, estando a baliza 2 entre as balizas 1 e 3.

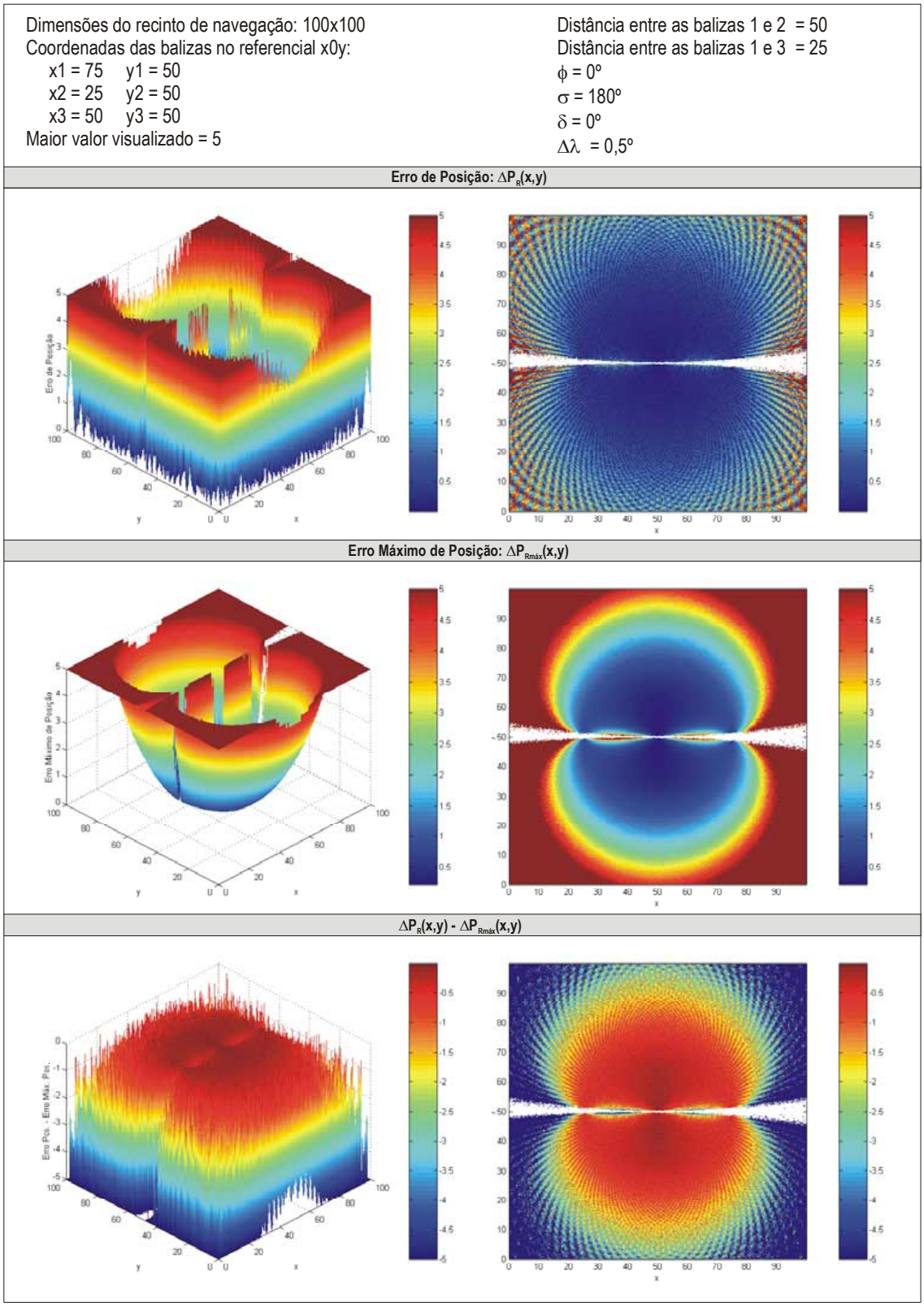
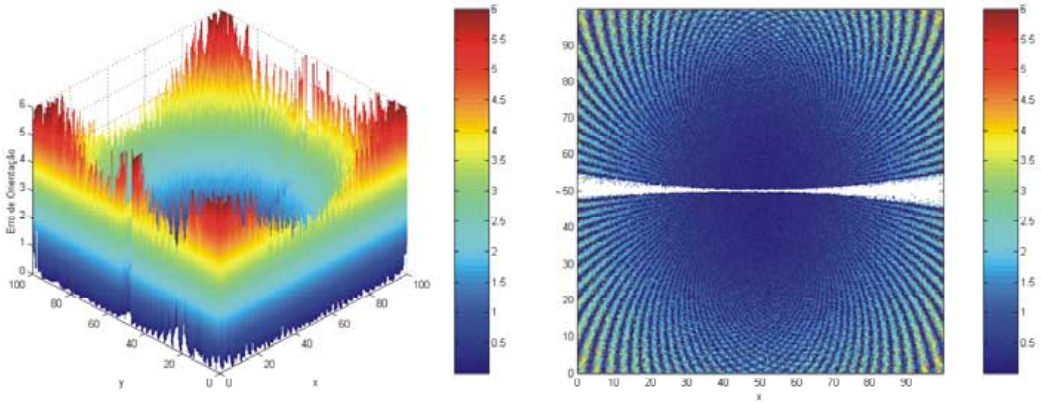


Figura 6.26: Erro de posição, erro máximo de posição e diferença entre o erro de posição e o erro máximo de posição, na região próxima de três balizas colineares, estando a baliza 3 entre as balizas 1 e 2.

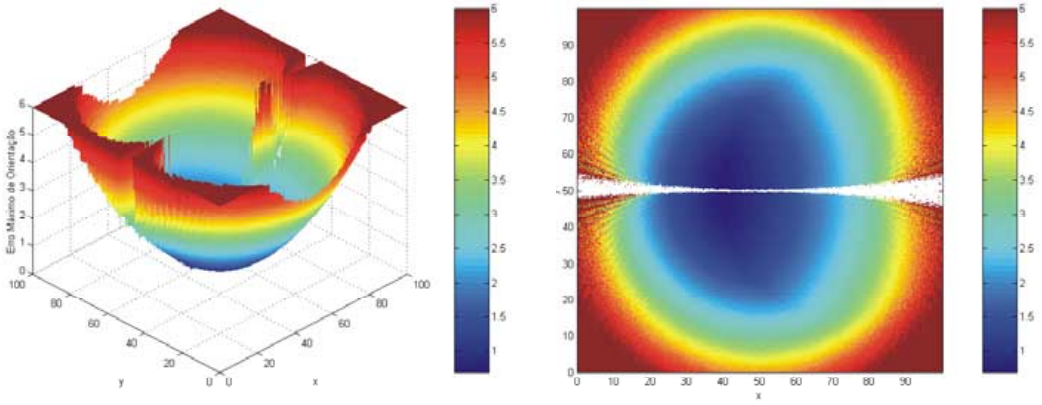
Dimensões do recinto de navegação: 100x100
Coordenadas das balizas no referencial x0y:
x1 = 75 y1 = 50
x2 = 25 y2 = 50
x3 = 50 y3 = 50
Maior valor visualizado = 6º

Distância entre as balizas 1 e 2 = 50
Distância entre as balizas 1 e 3 = 25
ϕ = 0º
σ = 180º
δ = 0º
$\Delta\lambda$ = 0,5º

Erro de Orientação: $\Delta\theta_R(x,y)$



Erro Máximo de de Orientação: $\Delta\theta_{Rmáx}(x,y)$



$\Delta\theta_R(x,y)$ - $\Delta\theta_{Rmáx}(x,y)$

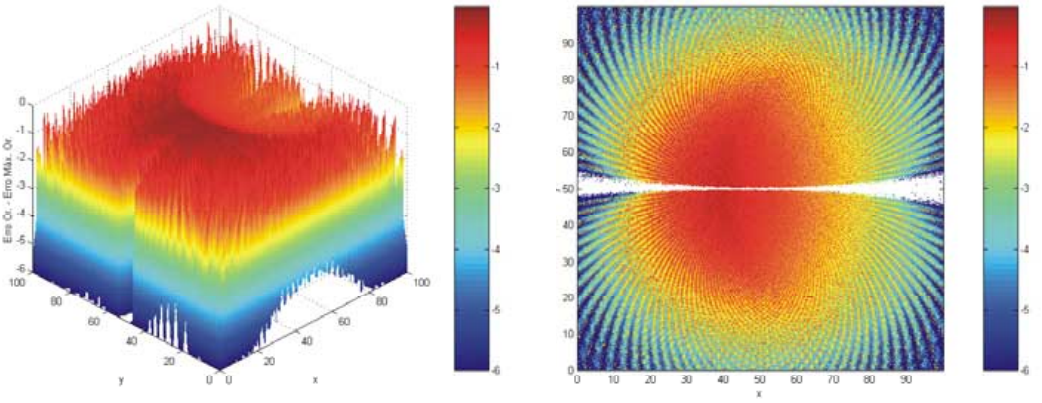


Figura 6.27: Erro de orientação, erro máximo de orientação e diferença entre o erro de orientação e o erro máximo de orientação, na região próxima de três balizas colineares, estando a baliza 3 entre as balizas 1 e 2.

6.3.4 Quarto Conjunto de Testes

O quarto conjunto de testes tem por objectivo determinar, para várias posições do robô relativamente às balizas, o tempo necessário para calcular a posição, a orientação e as incertezas que lhes estão associadas, quando se recorre ao Algoritmo Generalizado de Triangulação Geométrica e ao método de caracterização das incertezas de posição e de orientação proposto no Capítulo 5.

Utilizando a configuração de balizas representada na Figura 6.28 repetem-se os mesmos seis passos do anterior conjunto de testes (Figura 6.15) para posições de um robô pontual cobrindo quadrados de 1×1 unidades de comprimento, com incrementos de 0,001 unidades de comprimento tanto na direcção *x* como na direcção *y*, em várias zonas do plano de navegação. Em cada ponto, atribui-se à orientação do robô um valor aleatoriamente escolhido, superior a -180º e inferior ou igual a 180º.

Mede-se o tempo necessário para calcular a posição, a orientação e as incertezas que lhes estão associadas (passos III e IV representados na Figura 6.15) no total das iterações feitas em cada quadrado. Este tempo depende do modo como se executam os *bytecodes* produzidos pelo compilador de *Java 2*.

A implementação da *JVM* incluída na versão 1.3 para Windows do *Java 2 SDK, Standard Edition,* é a *Java HotSpot Client Virtual Machine.* Por defeito, com esta *JVM* os *bytecodes* são executados no modo *mixed-only:* o código começa por ser executado com um interpretador normal e, graças a um *compilador adaptativo,* os segmentos do código mais utilizados são compilados para *código nativo*. Os restantes *bytecodes* são executados por um interpretador de *bytecodes*. Também é possível executar um programa no modo *interpreted-only*, no qual todos os *bytecodes* são executados por um interpretador de *bytecodes*. Para um número suficientemente elevado de iterações, o tempo de execução de um programa no modo *mixed-only* é consideravelmente inferior ao obtido com modo *interpreted-only*.

Dividindo o tempo total pelo número de iterações realizadas em cada quadrado obtém-se o tempo médio de cada iteração (tempo necessário para executar uma vez os passos III e IV representados na Figura 6.15).

Na Tabela 6.5 indicam-se, para cada quadrado, os tempos médios por iteração correspondentes à execução no modo *interpreted-only* e no modo *mixed-only*. O número de iterações realizadas em cada teste assegura a execução das optimizações próprias do modo *mixed-only* (estas só começam a ser feitas a partir de um número mínimo de iterações) e também reduz para valores insignificantes os efeitos do erro de resolução finita inerente ao método utilizado para a contagem de tempos[11].

A análise dos resultados obtidos (Tabela 6.5) permite concluir que o tempo necessário para calcular em cada ponto a posição, a orientação e as incertezas que lhes estão associadas é da ordem de poucas décimas de milissegundo no modo *interpreted-only* e da ordem das dezenas de microssegundos no modo *mixed-only*. Se, em vez de *Java 2,* se recorrer a uma linguagem de programação compilada (por exemplo *C*), será certamente possível obter tempos ainda mais reduzidos.

Se houver mais de três balizas visíveis, os reduzidos tempos de cálculo tornam possível testar em tempo real várias configurações de balizas, e depois escolher os valores de posição e de orientação associados aos menores valores de de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. Com um tempo de cálculo de 150μs é possível analisar 10 configurações de balizas em apenas 1,5ms. Um veículo que se mova à velocidade de 1m/s – na Europa, esta é a máxima velocidade permitida para veículos automáticos que operam em ambientes onde há pessoas (Castleberry, 1991) – só consegue deslocar-se 1,5mm durante esse período de tempo.

Nas zonas E, F, G e H do plano de navegação é possível que $\Delta\tau_{máx}$ não ocorra num vértice da superfície de incerteza de posição, pelo que são efectuados mais cálculos. Isso explica os tempos superiores obtidos, no modo *interpreted-only*, nos quadrados pertencentes a essas zonas do plano. No modo *mixed-only* não é fácil interpretar as diferenças entre os tempos obtidos em cada zona, que dependem muito das optimizações inerentes a esse modo de executar os *bytecodes*.

[11] À contagem de tempos recorrendo ao método System.currentTimeMillis() está associado um erro de resolução finita que varia de acordo com a plataforma. Verificou-se experimentalmente que, na plataforma utilizada para executar os programas de teste, este erro é de cerca de 10ms.

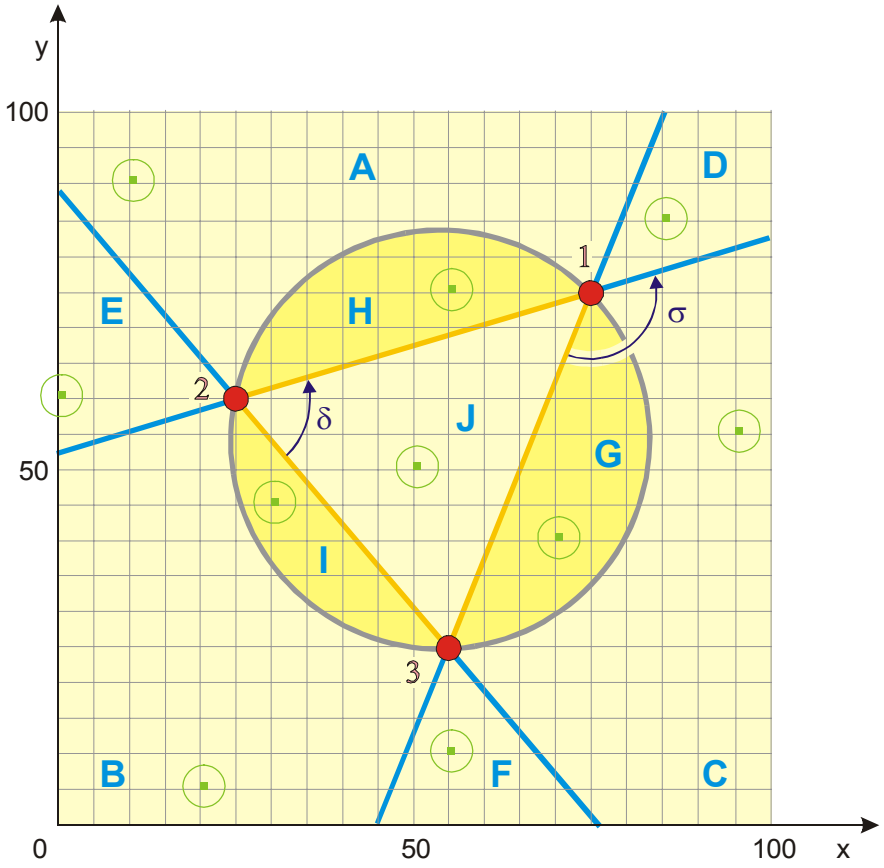


Figura 6.28: Configuração de balizas utilizada no quarto conjunto de testes. As regiões analisadas são os quadrados de 1 x 1 unidades de comprimento representados a verde, no interior das pequenas circunferências com a mesma cor.

Tabela 6.5: Resultados do quarto conjunto de testes.

Zona do plano de navegação	$x_{mín}$ $y_{mín}$	$x_{máx}$ $y_{máx}$	Número de iterações	Tempo médio de cada iteração no modo *interpreted-only* (ms)	Tempo médio de cada iteração no modo *mixed-only* (ms)
A	10 90	11 91	1001000	0,14	0,06
B	20 5	21 6	1000000	0,14	0,07
C	90 55	91 56	1001000	0,14	0,06
D	85 85	86 86	1000000	0,14	0,06
E	0 60	1 61	1001000	0,18	0,08
F	55 10	56 11	1002001	0,18	0,08
G	70 40	71 41	1001000	0,18	0,08
H	55 75	56 76	1001000	0,18	0,08
I	30 45	31 46	1001000	0,14	0,06
J	50 50	51 51	1002001	0,15	0,07

6.4 Conclusões

O Algoritmo Generalizado de Triangulação Geométrica está sujeito às restrições que são comuns a todos os algoritmos de autolocalização por triangulação. No entanto, as restrições específicas do Algoritmo de Triangulação Geométrica não se lhe aplicam. De facto, o algoritmo generalizado possui as seguintes características:

- As três balizas usadas pelo algoritmo podem ser aleatoriamente numeradas 1, 2 e 3 (não é necessário numerar as balizas consecutivamente no sentido directo).

- As três balizas podem ser colocadas em quaisquer pontos do plano de navegação (desde que duas balizas não partilhem a mesma posição).

- O ângulo formado pelos segmentos de recta que unem o robô às balizas 1 e 2 (λ_{12}) e o ângulo formado pelos segmentos de recta que unem o robô às balizas 1 e 2 (λ_{31}) podem ser iguais ou superiores a 180º. Em particular, é possível a autolocalização do robô quando este se encontra sobre a recta definida pelas balizas 1 e 2 ou a recta definida pelas balizas 1 e 3.

- O algoritmo funciona de forma consistente dentro, fora ou sobre o triângulo formado por três balizas não colineares (excepto nos pontos em que se aplica alguma das restrições comuns a todos os algoritmos de autolocalização por triangulação) ou fora da recta definida por três balizas colineares.

Apresentaram-se os resultados obtidos em quatro conjuntos de testes, realizados mediante a simulação por computador, cujos programas foram escritos em *Java 2*.

Com o primeiro conjunto de testes verificou-se que o Algoritmo Generalizado de Triangulação Geométrica funciona com balizas não colineares ordenadas no sentido directo ou no sentido inverso e com os três tipos de configurações de balizas colineares. Como se esperava, os erros de posição e de orientação são significativos quando o robô está sobre ou perto da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares.

Com o segundo conjunto de testes verificou-se de que modo é que os erros de medição dos ângulos afectam os erros de posição e de orientação, quando o robô se encontra próximo ou afastado das balizas. Eis algumas propriedades dos erros obtidos:

- Concordam com a análise feita previamente;
- São pequenos dentro do triângulo formado pelas três balizas;
- Aumentam significativamente à medida que o robô se aproxima da circunferência definida pelas três balizas;
- Decaem abruptamente quando o robô se afasta desta circunferência numa direcção radial e permanecem pequenos nas suas vizinhanças;
- Voltam a crescer, de forma mais suave, à medida que o robô se continua a afastar das balizas;
- Aumentam cerca de dez vezes quando a incerteza de medição de ângulos é multiplicada por dez.

Os resultados sugerem que, se a incerteza de medição de ângulos for suficientemente pequena, o robô será capaz de se localizar, cometendo apenas pequenos erros de localização, numa extensa área do plano de navegação.

O terceiro conjunto de testes validou o método de caracterização das incertezas de posição e de orientação que foi proposto no Capítulo 5. Este funcionou sempre correctamente em todos os testes realizados. Não houve uma única vez em que ΔP_R tenha sido superior a $\Delta P_{Rmáx}$ ou $\Delta\theta_R$ tenha sido superior a $\Delta\theta_{Rmáx}$.

Verificou-se que, em geral, a função $\Delta P_{Rmáx}(x,y)$ é uma boa envolvente da função $\Delta P_R(x,y)$, excepto nas imediações da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares. Aí acontece que os valores de $\Delta P_{Rmáx}(x,y)$ calculados em cada ponto ultrapassam largamente os respectivos valores de $\Delta P_R(x,y)$.

Verificou-se também que a função $\Delta\theta_{Rmáx}(x,y)$ só é tão boa envolvente da função $\Delta\theta_R(x,y)$ em algumas zonas do plano de navegação ou então se forem utilizadas configurações particulares de balizas. Os valores de $\Delta\theta_{Rmáx}(x,y)$ calculados nas imediações da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares ultrapassam largamente os respectivos valores de $\Delta\theta_R(x,y)$.

Os resultados obtidos no quarto conjunto de testes permitem concluir que o tempo necessário para calcular, em cada ponto em que se encontra o robô, a sua posição, a sua orientação e as incertezas que lhes estão associadas varia entre algumas dezenas de microssegundos e poucas décimas de milissegundo, dependendo do modo de execução do programa. Uma vez que este foi escrito em *Java 2*, é de esperar que se obtenham tempos de cálculo ainda mais reduzidos se se recorrer a uma linguagem de programação compilada, por exemplo ao *C*.

Se houver mais de três balizas visíveis, os reduzidos tempos de cálculo tornam possível testar dezenas de configurações de balizas em poucos milissegundos, podendo depois escolher-se os valores de posição e de orientação associados aos menores valores de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. No período de tempo correspondente à execução de todas estas operações, tendo em conta as velocidades máximas habitualmente praticadas por veículos automáticos, é de esperar que o veículo que se está a localizar só se consiga deslocar alguns milímetros.

O ideal teria sido obter também resultados experimentais correspondentes a cada um dos testes descritos. No entanto, a elaboração em tempo útil de um sistema que o permitisse iria requerer um trabalho de equipa que estava fora do âmbito previsto para a realização deste Doutoramento.

Como trabalho futuro, à luz dos resultados obtidos no terceiro conjunto de testes reforça-se a sugestão de que seja estudada a relação entre τ e λ_1, com o objectivo de tornar a função $\Delta\theta_{Rmáx}(x,y)$ uma melhor envolvente da função $\Delta\theta_R(x,y)$. Sugerem-se, ainda, as seguintes tarefas:

- Validar, com base em resultados experimentais, o Algoritmo Generalizado de Triangulação Geométrica e o novo método de caracterização das incertezas de posição e de orientação;

- Investigar a adaptação do Algoritmo Generalizado de Triangulação Geométrica à autolocalização absoluta, a três dimensões, por triangulação.

7. Conclusões e Sugestões de Trabalho Futuro

O primeiro objectivo do trabalho desenvolvido foi a eliminação das limitações específicas do Algoritmo de Triangulação Geométrica que, além de resolver o Problema de Pothenot, permite que um veículo determine a sua orientação. Este algoritmo tem sido considerado de menor interesse por só funcionar coerentemente dentro do triângulo definido por três balizas, sendo preterido em favor de outros que, por sua vez, também possuem limitações. Ficou demonstrado que, enquanto as limitações específicas destes últimos são inerentes aos próprios métodos, e por isso mesmo inultrapassáveis, o Algoritmo de Triangulação Geométrica pode ser generalizado por forma a conservar apenas as limitações que são comuns a qualquer algoritmo de autolocalização por triangulação. Como resultado da investigação realizada, desenvolveu-se o Algoritmo Generalizado de Triangulação Geométrica.

O segundo objectivo proposto foi o desenvolvimento de um método, aplicável ao Algoritmo Generalizado de Triangulação Geométrica, capaz de:

1. determinar em tempo real as incertezas associadas à posição e à orientação calculadas;

2. detectar situações nas quais a localização não é possível.

Realizou-se uma análise e classificação dos métodos de localização habitualmente utilizados na robótica móvel, em particular dos que recorrem a balizas, cujos resultados estão apresentados no Capítulo 2 e no Capítulo 3.

Procedeu-se, depois, ao estudo de vários algoritmos de autolocalização por triangulação com três balizas, que resolvem o Problema de Pothenot e são apresentados no Capítulo 4. Foi no decorrer desse estudo que surgiu a solução para a generalização do Algoritmo de Triangulação Geométrica. A versão generalizada deste algoritmo está apresentada no Capítulo 6.

Verificou-se que algumas das técnicas utilizadas na generalização do Algoritmo de Triangulação Geométrica podem também ser aplicadas a outros algoritmos de

autolocalização por triangulação com três balizas. A descrição dessas técnicas faz-se na primeira parte do quadro de análise proposto no Capítulo 5.

O desenvolvimento do método de caracterização de incertezas de posição e de orientação – cuja descrição constitui o resto do Capítulo 5 – foi a tarefa mais árdua deste trabalho. O caminho a seguir só foi encontrado depois de diversas tentativas de abordar o problema pelos métodos de análise de propagação de incertezas habitualmente utilizados. Estes envolvem o cálculo de derivadas parciais que, no caso da autolocalização por triangulação, são extremamente difíceis de obter. Surgiu, assim, a necessidade de fazer novas aproximações em métodos que, pela sua própria natureza, já eram aproximados. Como resultado dessas aproximações de validade duvidosa, ao fim de muitos meses de testes, os avanços conseguidos eram pequenos e o cálculo dos erros máximos continuava a falhar em alguns pontos do plano de navegação, nomeadamente na região próxima da circunferência definida por três balizas não colineares. Uma vez iniciado o caminho que levaria à solução definitiva, foram ainda inúmeras as dificuldades a vencer antes de a alcançar.

Recorrendo à simulação por computador, realizaram-se vários testes para analisar e validar o Algoritmo Generalizado de Triangulação Geométrica e também o método de cálculo dos erros máximos de posição e de orientação. Os resultados encontram-se no Capítulo 6.

Por fim, foi necessário consumir bastante tempo na revisão e na aquisição de alguns conceitos de estatística e análise numérica, na execução em *CorelDraw* (versão 9.337) das figuras que ilustram esta tese, e também na aprendizagem – ao nível do utilizador – das ferramentas de *software* necessárias à execução dos testes e à visualização gráfica dos resultados obtidos.

Seguidamente, no ponto 7.1 resumem-se as principais conclusões do trabalho desenvolvido e enumeram-se os contributos desta tese. No ponto 7.2 apresentam-se algumas sugestões de trabalho futuro.

7.1 Conclusões

Alguns dos métodos de localização analisados no Capítulo 2 são de localização absoluta: permitem determinar a posição e/ou a orientação sem recorrer a suposições

sobre movimentos anteriores do veículo que se pretende localizar. Os outros são de localização relativa. Vários métodos de localização absoluta – por exemplo, os que utilizam balizas – requerem que o ambiente seja preparado para efeitos de navegação.

É muito importante que um sistema de localização possua a capacidade de determinar a posição e a orientação de um robô móvel sem recorrer a suposições sobre movimentos anteriores uma vez que, em muitas situações, a informação sobre esses movimentos se pode perder ou não ser suficientemente fiável. Os métodos destinados à localização relativa de robôs móveis são simples de usar, não dependem de referências externas e actualizam medidas a frequências elevadas, mas são geralmente pouco exactos para assegurar a navegação por períodos de tempo longos ou grandes distâncias.

De todos os métodos estudados no Capítulo 2, a localização absoluta com balizas surge como a solução mais adequada ao desenvolvimento de um sistema fiável e de custo relativamente baixo para a localização contínua em tempo real de robôs móveis que navegam com velocidades de alguns metros por segundo, em ambientes (exteriores ou interiores) quase-estruturados e não muito obstruídos. A tecnologia actual, nomeadamente a dos sistemas ópticos com balizas activas, permite obter uma exactidão de medição de posição da ordem dos milímetros e uma exactidão de medição de orientação da ordem das centésimas de grau.

Entre os diversos métodos de trilateração e de triangulação com balizas analisados no Capítulo 3, a autolocalização absoluta por triangulação – baseada na resolução do Problema de Pothenot – surge como a mais indicada para localizar simultaneamente vários robôs que navegam a duas dimensões, desde que não ocorram inclinações significativas desses robôs. Este método pode ser implementado com sistemas passivos de localização e não requer a sincronização entre as balizas e o robô nem a sincronização entre balizas. É o único que permite que o robô determine a sua orientação recorrendo exclusivamente às medições efectuadas num mesmo instante e a partir de um só ponto. A posição do robô também pode ser calculada recorrendo exclusivamente a essas medições.

No Capítulo 4 apresentou-se uma definição do problema da autolocalização por triangulação. Abordou-se a ambiguidade de posição que consiste no facto de, a um dado conjunto de medidas de ângulos, corresponder mais do que uma posição possível no

plano de navegação. Apresentaram-se algumas das soluções habitualmente utilizadas para resolver este problema. Mostrou-se que todos os algoritmos de autolocalização baseados exclusivamente na triangulação estão sujeitos a duas restrições:

1. Um robô tem de "ver", pelo menos, três balizas para se poder localizar;

2. Um robô não se consegue localizar quando está sobre a circunferência definida por três balizas não colineares ou a recta definida por três balizas colineares.

Resultando destas duas restrições, há algumas linhas do plano de navegação nas quais a autolocalização por triangulação não é possível.

Analisaram-se vários algoritmos de triangulação que podem ser utilizados na autolocalização absoluta, a duas dimensões, de robôs móveis, com sistemas passivos de localização. Verificou-se que, além das duas restrições apontadas, cada um dos algoritmos estudados possui limitações específicas.

O Algoritmo de Triangulação Geométrica requer que as balizas sejam previamente ordenadas de uma forma específica. No entanto, a sua principal limitação reside no facto de só funcionar de forma coerente dentro do triângulo formado por três balizas. Este problema, que tem origem no modo como se definem os ângulos utilizados nos cálculos, é a razão pela qual o algoritmo tem sido considerado de menor interesse.

Vários autores têm preferido recorrer a algoritmos de triangulação baseados na intersecção de circunferências. No entanto, estes algoritmos possuem uma limitação que lhes é inerente e não pode ser eliminada: o raio da circunferência definida pelo robô e por duas balizas torna-se infinito quando as balizas ficam colineares com o robô (a circunferência degenera numa recta) e, nessas circunstâncias, a posição do robô não pode ser calculada. Assim, nos algoritmos que recorrem à intersecção de circunferências há duas ou três rectas do plano de navegação, cada uma delas definida por um par de balizas, ao longo das quais a localização não é possível.

À partida, o Algoritmo de Triangulação Geométrica é rápido, simples e versátil, uma vez que não recorre à resolução de sistemas de equações e não requer nenhuma particular configuração de balizas. Além disso, não requer estimativas iniciais de

posição ou de orientação, não produz soluções múltiplas e o problema da solução divergir não se coloca (não é um método iterativo). As características mencionadas justificam o trabalho realizado para eliminar as suas limitações específicas.

O quadro de análise da autolocalização absoluta por triangulação com três balizas, apresentado no Capítulo 5, constitui a base sobre a qual se fez a generalização do Algoritmo de Triangulação Geométrica e é aplicável a outros algoritmos de triangulação. Inclui um novo método para caracterizar em tempo real as incertezas associadas à posição e à orientação calculadas e para detectar situações nas quais a localização não é possível.

- Se se partir do princípio que os erros de medição têm limites finitos conhecidos, as incertezas de posição e de orientação podem ser caracterizadas por um *erro máximo de posição* $\Delta P_{Rmáx}$ e um *erro máximo de orientação* $\Delta\theta_{Rmáx}$. Apresentou-se um algoritmo para cálculo de $\Delta P_{Rmáx}$ e outro para cálculo de $\Delta\theta_{Rmáx}$.

- Se se considerar que os erros de medição possuem distribuições de probabilidade gaussianas, o algoritmo de cálculo de $\Delta P_{Rmáx}$ pode ser usado na caracterização da incerteza de posição devida a esses erros. O algoritmo permite calcular uma distância tal que existe uma probabilidade superior a um valor previamente estipulado de o erro de posição devido aos erros aleatórios de medição não ser superior a essa distância.

O novo método possui as seguintes características:

a) Funciona com todos os algoritmos de autolocalização absoluta por triangulação com três balizas (é independente do algoritmo de triangulação utilizado).

b) É exacto (não há aproximações que lhe sejam inerentes).

c) Por ser exacto, não requer que a incerteza de medição de ângulos se mantenha abaixo de um determinado limiar.

d) Não requer o cálculo de derivadas parciais com expressões analíticas difíceis de obter.

e) Adapta-se aos diferentes modos possíveis de medir ângulos.

f) Só deve ser utilizado se o algoritmo de triangulação usado no cálculo da posição for suficientemente rápido uma vez que, para cada posição calculada do robô, esse algoritmo tem de ser executado 5 ou 7 vezes, dependendo do modo como os ângulos são medidos.

g) Para que funcione correctamente, é necessário e suficiente que não se verifiquem certas condições, que foram apresentadas. Isto implica uma redução da superfície navegável, que é tanto maior quanto maior for a incerteza de medição dos ângulos.

O Algoritmo Generalizado de Triangulação Geométrica, apresentado no Capítulo 6, está sujeito às restrições que são comuns a todos os algoritmos de autolocalização por triangulação. No entanto, as restrições específicas do Algoritmo de Triangulação Geométrica não se lhe aplicam. De facto, o algoritmo generalizado possui as seguintes características:

- As três balizas usadas pelo algoritmo podem ser aleatoriamente numeradas 1, 2 e 3.

- As três balizas podem ser colocadas em quaisquer pontos do plano de navegação (desde que duas balizas não partilhem a mesma posição).

- O ângulo formado pelos segmentos de recta que unem o robô às balizas 1 e 2 (λ_{12}) e o ângulo formado pelos segmentos de recta que unem o robô às balizas 1 e 2 (λ_{31}) podem ser iguais ou superiores a 180º. Em particular, é possível a autolocalização do robô quando este se encontra sobre a recta definida por quaisquer duas balizas.

- O algoritmo funciona de forma consistente dentro, fora ou sobre o triângulo formado por três balizas não colineares (excepto nos pontos em que se aplica

alguma das restrições comuns a todos os algoritmos de autolocalização por triangulação) ou fora da recta definida por três balizas colineares.

Finalmente, apresentaram-se os resultados obtidos em quatro conjuntos de testes realizados mediante a simulação por computador:

- Com o primeiro conjunto de testes verificou-se que o Algoritmo Generalizado de Triangulação Geométrica funciona com balizas não colineares ordenadas no sentido directo ou no sentido inverso e com os três tipos de configurações de balizas colineares. Os erros de posição e de orientação são significativos quando o robô está sobre ou perto da circunferência definida por três balizas não colineares ou da recta definida por três balizas colineares.

- Com o segundo conjunto de testes verificou-se de que modo é que os erros de medição dos ângulos afectam os erros de posição e de orientação, quando o robô se encontra próximo ou afastado das balizas. Os resultados sugerem que, se a incerteza de medição de ângulos for suficientemente pequena, um robô será capaz de se localizar, cometendo apenas pequenos erros de localização, numa extensa área do plano de navegação.

- O terceiro conjunto de testes validou o método de cálculo das incertezas de posição e de orientação anteriormente proposto, que funcionou sempre correctamente.

- Os resultados obtidos no quarto conjunto de testes permitem concluir que o tempo necessário para calcular, em cada ponto em que se encontra o robô, a sua posição, a sua orientação e as incertezas que lhes estão associadas varia entre algumas dezenas de microssegundos e poucas décimas de milissegundo, dependendo do modo de execução do programa. Uma vez que este foi implementado em *Java 2*, é de esperar que se obtenham tempos de cálculo ainda mais reduzidos se se recorrer a uma linguagem de programação compilada, por exemplo ao *C*.

A obtenção de resultados experimentais correspondentes a cada um destes testes seria desejável. No entanto, a elaboração em tempo útil de um sistema que permitisse

obter tais resultados iria requerer um trabalho de equipa que estava fora do âmbito previsto para a realização deste Doutoramento.

Em resumo, os principais contributos científicos desta tese são os seguintes:

- Um quadro de análise da autolocalização absoluta por triangulação com três balizas, que inclui:

  a) uma cuidadosa definição de ângulos a utilizar em algoritmos de triangulação;

  b) uma nova especificação do problema da autolocalização absoluta, a duas dimensões, por triangulação;

  c) um novo método para caracterizar em tempo real as incertezas associadas à posição e à orientação calculadas e para detectar situações nas quais a localização não é possível. Pode ser usado quer se parta do princípio que os erros de medição têm limites finitos conhecidos quer se considere que esses erros possuem distribuições de probabilidade gaussianas (no segundo caso, o método permite caracterizar apenas a incerteza de posição).

- A generalização do Algoritmo de Triangulação Geométrica, feita com base no quadro de análise proposto.

Além disso, no Capítulo 3 sugeriram-se dois novos métodos que permitem a autolocalização recorrendo apenas a duas balizas:

- O primeiro método requer que, a partir de um robô que se pretende autolocalizar, se meçam simultaneamente um ângulo orientado formado pelos segmentos que unem o robô a duas balizas e a distância a uma delas. Possui a complexidade inerente à medição simultânea de ângulos e distâncias.

- O segundo método requer que, a partir de um robô que se pretende autolocalizar, se meçam simultaneamente um ângulo orientado formado pelos segmentos que unem o robô a duas balizas e a diferença das distâncias do robô a cada uma dessas balizas. Possui a complexidade inerente à medição simultânea de ângulos e diferenças de distâncias.

Em ambos os métodos, uma vez calculada a posição do robô, a sua orientação pode determinar-se a partir da medição do ângulo orientado formado por um semieixo de referência fixo no robô com o segmento de recta que une o robô a uma das balizas.

No Capítulo 4 deram-se, ainda, os seguintes contributos:

- Sugeriu-se um algoritmo de triangulação simples, mas que consiste na resolução de um sistema de três equações não lineares (os algoritmos de outros autores, apresentados nos pontos 4.5 a 4.12, são mais complexos mas envolvem cálculos mais fáceis de executar).

- Sugeriu-se uma especificação do Algoritmo de Triangulação Baseado na Pesquisa Iterativa.

- Sugeriu-se uma especificação do Algoritmo de Triangulação Baseado no Método de Newton-Raphson.

7.2 Sugestões de Trabalho Futuro

Como trabalho futuro, sugerem-se as seguintes tarefas:

- Validar, com base em resultados experimentais, o Algoritmo Generalizado de Triangulação Geométrica e o novo método de caracterização das incertezas de posição e de orientação.

- Investigar uma generalização da análise desenvolvida no capítulo 5 que seja adequada à autolocalização absoluta, a três dimensões, por triangulação.

- Investigar a adaptação do Algoritmo Generalizado de Triangulação Geométrica à autolocalização absoluta, a três dimensões, por triangulação.

- Estudar de forma mais aprofundada as propriedades geométricas dos gráficos que representam, num referencial ortonormado $\lambda_{12}0\lambda_{31}$, os pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação, para uma dada configuração de balizas, com o fim de obter uma mais completa caracterização dessa configuração.

- Estudar a relação entre os ângulos usados no cálculo da orientação do robô, tendo em vista produzir uma melhor estimativa do erro máximo de orientação.

- Investigar a possibilidade de se usar o algoritmo desenvolvido para determinar o erro máximo de orientação na caracterização da incerteza de orientação devida aos erros aleatórios de medição.

- Investigar a possibilidade de se simplificar os algoritmos desenvolvidos para determinar os erros máximos de posição e de orientação.

- Investigar os dois métodos de autolocalização propostos no Capítulo 3, os quais permitem a autolocalização recorrendo apenas a duas balizas.

- Elaborar um algoritmo capaz de escolher, entre todas as balizas visíveis de uma dada posição, o terno de balizas e a respectiva ordenação que permitem calcular a posição e a orientação do robô com valores mínimos de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$.

  Uma solução óbvia é recorrer aos algoritmos desenvolvidos e calcular os valores de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$ que se podem obter com cada um dos ternos de balizas disponíveis e com cada uma das ordenações possíveis, escolhendo depois o terno e a ordenação que geraram os menores valores de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. Mas talvez seja possível conhecer *a priori* (ou seja, antes de calcular a posição e a orientação) quais são, num dado ponto do plano de navegação, o terno de balizas e a ordenação que irão originar esses valores mínimos de $\Delta P_{Rmáx}$ e $\Delta\theta_{Rmáx}$. Pode ser que parte da solução se encontre nas propriedades

geométricas dos gráficos que representam, num referencial ortonormado $\lambda_{12}0\lambda_{31}$, os pontos correspondentes aos pares de valores (λ_{12}, λ_{31}) obtidos em cada posição do plano de navegação, para uma dada configuração de balizas.

- Desenvolver um método para determinar qual é a configuração de balizas que minimiza a fracção de um dado recinto de navegação na qual ocorrem erros de posição e de orientação superiores a um valor máximo admissível.

É possível que as soluções para os problemas referidos nas duas últimas tarefas propostas estejam parcialmente radicadas no algoritmo de triangulação utilizado. Mas talvez se consiga chegar a algumas conclusões gerais, aplicáveis a todos os algoritmos de autolocalização absoluta por triangulação com três balizas.

Referências

Abreu *et al.*, 1994

Abreu, M. C.; Matias, L. e Peralta, L. F.; Física Experimental – Uma Introdução, Editorial Presença, 1994.

Adams, 2002

Adams, Thomas M.; A2LA Guide for the Estimation of Measurement Uncertainty in Testing; A2LA, Julho de 2002.

AGVE, 2000

AGV Electronics; Laser Navigation; Novembro de 2000.
http://www.agve.se

AGVE, 2001a

AGV Electronics; Inductive Guidance; Junho de 2001.
http://www.agve.se

AGVE, 2001b

AGV Electronics; Magnet-Gyro Guidance; Junho de 2001.
http://www.agve.se

AGVP, 2003a

AGV Products, Inc; Inertial Guidance; 2003.
http://www.agvp.com

AGVP, 2003b

AGV Products, Inc; Laser Guidance; 2003.
http://www.agvp.com

AGVP, 2003c

AGV Products, Inc; Wire Guidance; 2003.

http://www.agvp.com

Aider *et al.*, 2002

Aider, Omar Ait; Hoppenot, Philippe e Colle, Etienne; A Model to Image Straight Line Matching Method for Vision-Based Indoor Mobile Robot Self-Localization; Proceedings of the 2002 IEEE/RSJ International Conference on Robots and Systems, p. 460-465, EPFL, Lausanne, Suíça, Outubro de 2002.

Almeida, 1997

Almeida, Guilherme: Sistema Internacional de Unidades (SI), Grandezas e Unidades Físicas – Terminologia, Símbolos e Recomendações, Plátano Editora, 1997.

Araújo, 1998

Araújo, Paulo Ventura; Curso de Geometria; Gradiva, 1998.

Arras, 1998

Arras, Kai Oliver; An Introduction to Error Propagation: Derivation, Meaning and Examples of Equation $C_Y=F_X C_X F_X^T$ (Relatório Técnico nº EPFL-ASL-TR-98-01 R3); Autonomous Systems Lab, Institute of Robotic Systems, Swiss Federal Institute of Technology Lausanne, Setembro de 1998.

Ashokaraj *et al.*, 2003

Ashokaraj, I; Tsourdos, A.; White, B. e Silson, P.; Robot Localization Using Interval Analysis; 2003 IEEE International Conference on Sensors, p. 30-35, 2003.

Barney, 1988

Barney, George C.; Intelligent Instrumentation: Microprocessor Applications in Measurement and Control (2ª edição); Prentice Hall, 1988.

Berger e Kubitz, 1996

Berger, Mathias Oliver; Kubitz, Olaf; Application of Wireless Communication in Robotics; Aachen University of Technology, 1996.

Betke e Gurvits, 1997

Betke, Margrit; Gurvits, Leonid; Mobile Robot Localization Using Landmarks; IEEE Transactions on Robotics and Automation, Vol. 13, Nº 2, Abril de 1997.

Bettencourt, 1972

Bettencourt, Tedeschi; Navegação (Aeronáutica); Enciclopédia Luso Brasileira de Cultura, Vol. 13º, p. 1756-1760, Editorial Verbo, 1972.

Borenstein, 1994

Borenstein J.; Internal Correction of Dead-reckoning Errors With the Smart Encoder Trailer; International Conference on Intelligent Robots and Systems (IROS'94) – Advanced Robotic Systems and the Real World, pp 127-134, Munique, Alemanha, 12 a 16 de Setembro de 1994.

Borenstein e Feng, 1994

Borenstein, Johann e Feng, Liquiang; UMBmark – A Method for Measuring, Comparing, and Correcting Dead-reckoning Errors in Mobile Robots (Relatório Técnico), The University of Michigan UM-MEAM-94-22, Dezembro de 1994.

Borenstein e Feng, 1995

Borenstein, Johann e Feng, Liquiang; UMBmark – A Bechmark Test for Measuring Odometry Errors in Mobile Robots; 1995 SPIE Conference on Mobile Robots, Filadélfia, 22 a 26 de Outubro de 1995.

Borenstein e Feng, 1996a

Borenstein, Johann e Feng, Liquiang; Gyrodometry: A New Method for Combining Data from Gyros and Odometry in Mobile Robots; Proceedings of the 1996 IEEE International Conference on Robotics and Automation, pp. 423-428, Minneapolis, 22 a 28 de Abril de 1996.

Borenstein e Feng, 1996b

Borenstein, Johann e Feng, Liquiang; Measurement and Correction of Systematic Odometry Errors in Mobile Robots; IEEE Transactions on Robotics and Automation, Vol. 12, No 5, Outubro de 1996.

Borenstein *et al.*, 1996

Borenstein, J., Everett, H. R. e Feng, L.: Where am I? Sensors and Methods for Mobile Robot Positioning (Relatório Técnico), The University of Michigan, 1996.

Borenstein *et al.*, 1997

Borenstein, J; Everett, H. R.; Feng, L.; Wehe, D.; Mobile Robot Positioning - Sensors and Techniques, Journal of Robotic Systems, Special Issue on Mobile Robots. Vol. 14 No. 4, pp. 231 – 249, 1997.

Braasch e Dierendonck, 1999

Braasch, Michael S. e Dierendonck, A. J. Van; GPS Receiver Architectures and Measurements; Proceedings of the IEEE, Volume 87, Nº 1, Janeiro de 1999.

Bretz, 2000

Bretz, Elisabeth A.; X Marks the Spot, Maybe; IEEE Spectrum, Abril de 2000.

Briechle e Hanebeck, 2004

Briechle, K. e Hanebeck, U. D.; Localization of a Mobile Robot Using Relative Bearing Measurements; IEEE Transactions on Robotics and Automation, Vol. 20, Nº. 1, p. 36-44, Fevereiro de 2004.

Brown e Hwang, 1997

Brown, Robert Grover e Patrick Y. C.; Introduction to Random Signals and Applied Kalman Filtering (3ª ed.); John Wiley & Sons, Inc., 1997.

Campione *et al.*, 2001

Campione, Mary; Walrath, Kathy; Huml, Alison; The Java Tutorial, Third Edition – A Short Course on The Basics; The Java Series, Sun Microsystems; Addison-Wesley, 2001.

Castleberry, 1991

Castleberry, Guy A.; The AGV Handbook – A Handbook for the Selection of Automated Guided Vehicle Systems; AGV Decisions, Incorporated, 1991.

Cauchois *et al.*, 2002

Cauchois, Cyril; Brassart, Eric; Marhic, Bruno e Drocourt, Cyril; An Absolute Localization Method using Synthetic Panoramic Image Base; Proceedings of the OMNIVIS'02 - Third Workshop on Omnidirectional Vision, p. 128-135, 2 de Junho de 2002.

Christou *et al.*, 1992

Christou, N.; Parthenis, K.; Dimitriadis, B.; Gouvianakis, N.; Digital Models For Autonomous Vehicle Terrain-Following; Tzafestas, S. G. (ed.), Robotic Systems – Advanced Techniques and Applications, p. 407-414; Kluwer Academic Publishers, 1992.

Clapham, 1996

Clapham, Christopher; The Concise Oxford Dictionary of Mathematics (2ª ed.); Oxford University Press, 1996.

Clerentin *et al.*, 2002

Clerentin, Amaud; Delahoche, Laurent; Brassart, Eric e Cauchois, Cyril; Mobile Robot Localization Based on Multi Target Tracking; Proceedings of the 2002 IEEE International Conference on Robotics and Automation, p. 13-18, Washington, DC, Maio de 2002.

Cohen e Koss, 1992

Cohen, Charles; Koss, Frank V.; A Comprehensive Study of Three Object Triangulation; SPIE Vol. 1831, Mobile Robots VII, 1992.

Colon e Baudoin, 1996

Colon, Eric e Baudoin, Yves; Development and Evaluation of Distributed Control Algorithms for the Mobile Robot Nomad200; Mobile Robots XI and Automated Vehicle Control Systems - SPIE Proceedings, Volume 2903, 1996.

Costa *et al.*, 2004

Costa, Al; Kantor, George e Choset, Howie; Bearing-only Landmark Initialization with Unknown Data Association; Proceedings of the 2004 IEEE International

Conference on Robotics and Automation, p. 1764-1770, New Orleans, LA, Abril de 2004.

Crowley, 1995

Crowley, James L.; Mathematical Foundations of Navigation and Perception For an Autonomous Mobile Robot; Worshop on Reasoning With Uncertainty in Robotics; University of Amsterdam, Holanda; 4 a 6 de Dezembro de 1995.

Curtis, 1989

Curtis, S.; Transponder Technologies, Applications and Benefits; Colloquium on The Use of Electronic Transponders in Automation; Professional Group C6, 15 de Fevereiro de 1989.

Davis *et al.*, 1981

Davis, Raymond E.; Foote, Francis S.; Anderson, James M.; Mikhail, Edward M.; Surveying: Theory and Practice (6ª ed.); McGraw-Hill, 1981.

De Cecco, 2002

De Cecco, Mariolino; Self-Calibration of AGV Inertial-Odometric Navigation Using Absolute-Reference Measurements; IEEE Insrumentation and Measurement Technology Conference, p. 1513-1517, Anchorage, AK, USA, 21 a 23 de Maio de 2002.

Di Marco *et al.*, 2000

Di Marco, M.; Garulli, A.; Lacroix, S. e Vicino, A.; A Set Theoretic Approach to the Simultaneous Localization and Map Building Problem; Proceedings of the 39th IEEE Conference on Decision and Control, p. 833-838, Sidney, Austrália, Dezembro de 2000.

Dierendonck, 2001

Dierendonck, A. J. Van; Future GPS Civil Signals; Rocky Mountain Section of the Institute of Navigation meeting, Air Force Academy Officer's Club, Colorado Springs, 25 de Janeiro de 2001.

Diggelen e Abraham, 2001

Diggelen, Frank van e Abraham, Charles; Indoor GPS Technology; CTIA Wireless-Agenda, Dallas, Maio de 2001.

DLP, 1994

Dicionário da Língua Portuguesa (7ª ed.) , Porto Editora, 1994.

DM, 2003

Danaher Motion; Laser Scanner 4 – 2.0 – Technical Data; Buyer's Guide, 2003.

DoD, 2001

2001 Federal Radionavigation Systems; U. S. Department of Defense e U. S. Department of Transportation, 2001.

Dorrie, 1965

Dorrie, Heinrich; 100 Great Problems of Elementary Mathematics: Their History and Solution (trad. David Antin); Dover Publications, New York, 1965.

Dougherty e Giardina, 1988

Dougherty, Edward R. e Giardina, Charles R.; Mathematical Methods for Artificial Intelligence and Autonomous Systems; Prentice-Hall International, Inc., 1988.

Drane e Rizos, 1998

Drane, Chris; Rizos, Chris; Positioning Systems in Intelligent Transportation Systems; Artech House, 1998.

EGEMIN, 2002a

Egemin International; Camera Navigation; 2002.
http://www.egemin.com/

EGEMIN, 2002b

Egemin International; Laser Navigation; 2002.
http://www.egemin.com/

EGEMIN, 2002c

Egemin International; Magnet Navigation; 2002.
http://www.egemin.com/

EGEMIN, 2002d

Egemin International; Optical Navigation; 2002.
http://www.egemin.com/

EGEMIN, 2002e

Egemin International; Wire Navigation; 2002.
http://www.egemin.com/

Enge e Misra, 1999

Enge, Per e Misra, Pratap; Scanning the Issue/Technology: Special Issue on Global Positioning System; Proceedings of the IEEE, Volume 87, Nº 1, Janeiro de 1999.

Espartel, 1980

Espartel, Lélis; Curso de Topografia (7ª ed.); Editora Globo, 1980.

Everett, 1995

Everett, H.R.; Sensors for Mobile Robots; A K Peters, Junho de 1995.

EUROCONTROL, 1998

EUROCONTROL Standard Document for Area Navigation Equipment Operational Requirements and Functional Requirements (ed. 2.2); EUROCONTROL – European Organization for the Safety of Air Navigation, 1998.

Fisher e Ghassemi, 1999

Fisher, Steven C. e Ghassemi, Kamran; GPS IIF – The Next Generation; Proceedings of the IEEE, Volume 87, Nº 1, Janeiro de 1999.

Fishwick, 1994

Fishwick, P. A.; Computer Simulation: Growth Through Extension; Proceedings of European Simulation Multiconference, Barcelona, Espanha, 1994.

Fontana *et al.*, 2001a

Fontana, Richard D.; Cheung, Wai e Stansell, Tom; The Modernized L2 Civil Signal - Leaping Forward in the 21st Century; GPS World, Setembro de 2001.

Fontana *et al.*, 2001b

Fontana, Richard D.; Cheung, Wai; Novak, Paul M. e Stansell, Tom; The New L2 Civil Signal; ION GPS 2001 - 14th International Technical Meeting of the Satellite Division of the Institute of Navigation, Salt Lake City, Utah, 11 a 14 de Setembro de 2001.

Frappier *et al.*, 1992

Frappier, G.; Lemarquand, P. e Van den Bogaert, T.; Navigation And Perception Approach Of PANORAMA Project; Tzafestas, S. G. (ed.), Robotic Systems – Advanced Techniques and Applications, p. 391-398; Kluwer Academic Publishers, 1992.

Fuentes *et al.*, 1995

Fuentes, O.; Karlsson, J.; Meira, W.; Rao, R.; Riopka, T.; Rosca, J.; Sarukkai, R.; Van Wie, M.; Zaki, M.; Becker, T.; Frank, R.; Miller, B. e Brown, C. M.; Mobile Robotics 1994 (Relatório Técnico nº 588); Computer Science Department, The University of Rochester, Junho de 1995.

García-Tejero, 1981

García-Tejero, Francisco Domínguez; Topografia Abreviada (5ª ed.); Dossat, Madrid, 1981.

Garulli e Vicino, 2001

Garulli, Andrea e Vicino, Antonio; Set Membership Localization of Mobile Robots via Angle Measurements; IEEE Transactions on Robotics And Automation, Vol. 17, Nº. 4, p. 450-463, Agosto de 2001

Gning e Bonnifait, 2004

Gning, A. e Bonnifait, Ph.; Guaranteed Dynamic Localization using Constraints Propagation Techniques on Real Intervals; Proceedings of the 2004 IEEE International Conference on Robotics and Automation, p. 1499-1504, New Orleans, LA, Abril de 2004.

Grejner-Brzezinska, 2002

Grejner-Brzezinska, Dorota; GPS Instrumentation Issues; Manual of Geospatial Science and Technology; Bossler, J., Jenson, J., Mcmaster, R., & Rizos, C. (eds.), Taylor & Francis Inc., 2002.

Gu *et al.*, 2002

Gu, Jason; Meng, Max; Cook, Al e Liu, Peter X.; Sensor Fusion in Mobile Robot: Some Perspectives; Proceedings of the 4th World Congress on Intelligent Control and Automation, p. 1194-1199, Xangai, R. P. China, 10 a 14 de Junho de 2002.

Gutmann, 2002

Gutmann, Jens-Steffen; Markov-Kalman Localization for Mobile Robots; ICPR'02 – 16th International Conference on Pattern Recognition, Vol. 2, Quebec, Canadá, 11 a 15 de Agosto de 2002.

Hager *et al.*, 1993

Hager, Gregory D.; Engelson, Sean P. e Atiya, Sami; On Comparing Statistical and Set-Based Methods in Sensor Data Fusion; Proceedings of the IEEE International Conference on Robotics and Automation, p. 352 – 358, Atlanta, GA, 2 a 6 de Maio de 1993.

Hayet *et al.*, 2002

Hayet, Jean-Bernard; Esteves, Cláudia; Devy, Michel e Lerasle, Frédéric; Qualitative Modeling of Indor Environents from Visual Landmarks and Range Data; Proceedings of the 2002 IEEE/RSJ International Conference on Robots and Systems, p. 631-636, EPFL, Lausanne, Suíça, Outubro de 2002.

Hebert, 2000

Hebert, Martial; Active and Passive Range Sensing for Robotics; Proceedings of the 2000 IEEE International Conference on Robotics and Automation, p. 102-110, San Francisco, Califórnia, Abril de 2000.

Hernández *et al.*, 2003

Hernández, Sergio; Morales, Carlos A.; Torres, Jesus M. e Acosta, Leopoldo; A New Localization System for Autonomous Robots; Proceedings of the 2003 IEEE International Conference on Robotics and Automation, p. 1588-1593, Taipei, Taiwan, 14 a 19 de Setembro de 2003.

Hurn, 1989

Hurn, Jeff; GPS - A Guide to the Next Utility; Trimble Navigation, 1989.

ICD-200, 2000

ICD GPS 200C - Interface Control Document: Navstar GPS Space Segment / Navigation User Interfaces; ARINC Research Corporation, El Segundo, CA, USA, 1993 – 2000.

Jang *et al.*, 2002

Jang, Gijeong; Kim, Sungho; Lee, Wangheon e Kweon, Inso; Color Landmark Based Self-Localization for Indoor Mobile Robots; Proceedings of the 2002 IEEE International Conference on Robotics and Automation, p. 1037-1042, Washington, DC, Maio de 2002.

Jaulin *et al.*, 2002

Jaulin, Luc; Kieffer, Michel; Walter, Eric e Meizel, Dominique; Guaranteed Robust Nonlinear Estimation With Application to Robot Localization; IEEE Transactions on Systems Man, and Cybernetics – Part C: Applications and Reviews, Vol. 32, Nº 4, p.374-381, Novembro de 2002.

Ji *et al.*, 2003

Ji, Junhong; Indiveri, Giovanni; Ploeger, Paul-Gerhard; Bredenfeld, Ansgar; An Omni-Vision based Self-Localization Method for Soccer Robot; Proceedings of

the IEEE IV2003, Intelligent Vehicles Symposium, Columbus, Ohio, USA, 9 a 11 de Junho de 2003.

Jones *et al.*, 1999

Jones, Joseph L.; Seiger, Bruce A.; Flynn, Anita M.; Mobile Robots – Inspiration to Implementation (2ª ed.); A. K. Peters, Ltd., 1999.

Jong, 2002

Jong, Kees de; Future GPS and Galileo signals – Unprecedented Accuracy and Availability; GeoInformatics, Setembro de 2002

Kahan, 1996

Kahan, W; Lecture Notes on the Status of IEEE Standard 754 for Binary Floating-Point Arithmetic, 1996.

Kang e Jo, 2003

Kang, Hyun-Deok e Jo, Kang-Hyun; Self-Localization of Mobile Robot Using Omni-Directional Vision; Proceedings of the KORUS 2003 – 7th Korea-Russia International Symposium, p.86-91, 2003.

Kaplan, 1996

Kaplan, Elliot D.; Understanding GPS – Principles and Applications; Artech House, 1996.

Kelly, 1996

Kelly, Alonzo; 16-899A Mobile Robot Systems Course (Lecture Notes); The Robotics Institute, School of Computer Science, Carnegie Mellon University, 1996.

Kleeman, 2003

Kleeman, Lindsay; Advanced Sonar and Odornetry Error Modeling for Simultaneous Localisation and Map Building; Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, p. 699-704, Las Vegas, Nevada, Outubro de 2003.

Kortenkamp *et al.*, 1998

Kortenkamp, David; Bonasso, R. Peter; Murphy, Robin; Artificial Intelligence and Mobile Robots – Case Studies of Successful Robot Systems; American Association for Artificial Intelligence / AAAI Press / The MIT Press; 1998.

Kuipers e Levitt, 1988

Kuipers, Benjamin J. e Levitt, Tod S.; Navigation and Mapping in Large-Scale Space; AI Magazine, Vol. 9, nº 2, p. 25-43 Julho/Agosto de 1988.

Lee *et al.*, 2003

Lee, Dongheui; Chung, Woojin; Kim, Munsang; A Reliable Position Estimation Method of the Service Robot by Map Matching; Proceedings of the 2003 IEEE International Conference on Robotics and Automation, p. 2830-2835, Taipei, Taiwan, 14 a 19 de Setembro de 2003.

Leonard e Durrant-White, 1992

Leonard, John J.; Durrant-White, Hugh F.; Directed Sonar Sensing for Mobile Robot Navigation; Kluwer Academic Publishers, 1992.

Lin e Tummala, 1997

Lin, Cheng-Chih; Tummala, R. Lal; Mobile Robot Navigation Using Artificial Landmarks; Journal of Robotic Systems, 14(2), p. 93-106, 1997.

μ-BLOX, 2001

μ-BLOX AG; The GPS Dictionary – Acronyms, Abbreviations and Glossary related to GPS; 8 de Março de 2001.
http://www.u-blox.com

Marques, 2001

Marques, Manuel Rebelo; Navegar (2ªed.); Publicações Europa-América, 2001.

Marques *et al.*, 1996

Marques, Lino; Nunes, Urbano; Almeida, A. T. de; "Laser Range Finder" Para Robótica Móvel; Anais da Engenharia e Tecnologia Electrotécnica, Ano I, nº 1,

Junho de 1996.

Martins, 2001

Martins, F. Mário; Programação Orientada aos Objectos em Java 2 (4ªed.); Colecção Tecnologias de Informação; FCA – Editora de Informática, LDA., 2001.

Mcgillem e Rappaport, 1989

Mcgillem, Clare D. e Rappaport, Theodore S.; A Beacon Navigation Method for Autonomous Vehicles; IEEE Transactions on Vehicular Technology, Vol. 38, nº 3, Agosto de 1989.

McKerrow , 1991

McKerrow, P. J.; Introduction to Robotics; Addison-Wesley Publishing Company, Inc.; 1991.

Meizel *et al.*, 2002

Meizel, Dominique; Lévêque, Olivier; Jaulin, Luc e Walter, Eric; Initial Localization by Set Inversion; IEEE Transactions on Robotics and Automation, Vol. 18, Nº 6, p. 966-971, Dezembro de 2002.

MN, 1989

Manual de Navegação – Cálculos Náuticos (4ªed.); Instituto Hidrográfico, Lisboa, 1989.

Moody, 1971

Moody, Alton B.; Dead Reckoning; McGraw-Hill Encyclopedia of Science and Technology, Vol. 4, p. 26-29, McGraw-Hill, 1971.

NCG, 1996

Numerical Computation Guide; Sun Microsystems, 1996.

NDC, 1998

NDC Automation Inc.; Advantage Lazerway™! - Laser Guidance vs. Inertial Guidance (Technology Brief); USA (980129), 1998.

Novais, 1981

Novais, Maria Helena; Cálculo Vectorial e Geometria Analítica (2ª ed.); Dinalivro, 1981.

Owen e Nehmzow, 1998

Owen, Carl e Nehmzow, Ulrich; Landmark-based Navigation for a Mobile Robot; Proc. Simulation of Adaptive Behaviour '98, MIT Press, 1998.

Piskounov, 1997

Piskounov, N.; Cálculo Diferencial e Integral, Vol. II (11ª Ed.), Edições Lopes da Silva, 1997.

Prasser e Wyeth, 2003

Prasser, David e Wyeth, Gordon; Probabilistic Visual Recognition of Artificial Landmarks for Simultaneous Localization and Mapping; Proceedings of the 2003 IEEE International Conference on Robotics and Automation, p. 1291-1296, Taipei, Taiwan, 14 a 19 de Setembro de 2003.

Reed e James, 1997

Reed, Jeff H. e James, Robert D.; Position Location: Technology Overview and Business Opportunities; Wireless Opportunities Workshop, 22 de Outubro de 1997.

Rizos e Satirapod, 2001

Rizos, C. e Satirapod, C.; Differential GPS: How good is it now?; Measure & Map, 12, p. 19-21, 2001.

Saeedi *et al.*, 2003

Saeedi, P.; Lowe, D. G. e Lawrence, P. D.; 3D Localization and Tracking in Unknown Environments; Proceedings of the 2003 IEEE International Conference on Robotics and Automation, p. 1297-1303, Taipei, Taiwan, 14 a 19 de Setembro de 2003.

Satirapod *et al.*, 2001

Satirapod, C.; Rizos, C. e Wang, J.; GPS single point positioning with SA off: How accurate can we get?; Survey Review, 36(282), 255-262, 2001.

Schreiber e Dickerson, 1996

Schreiber, Michael J.; Dickerson, Stephen L.; Outdoor Tracking Using Machine Vision, Xenon Strobe Illumination, and Retroreflective Landmarks; Mobile Robots XI and Automated Vehicle Control Systems - SPIE Proceedings, Volume 2903, 1996.

Sena Esteves, 1996

Sena Esteves, João; O Uso de Transponders de Baixa Frequência na Identificação Automática (trabalho de síntese realizado no âmbito da prestação de Provas de Capacidade Científica); Guimarães, 1996.

Sena Esteves *et al.*, 2003

Sena Esteves, João; Carvalho, Adriano; Couto, Carlos; Generalized Geometric Triangulation Algorithm for Mobile Robot Absolute Self-Localization; ISIE 2003 - 2003 IEEE International Symposium on Industrial Electronics, Rio de Janeiro, Brasil, 9 a 12 de Junho de 2003.

Shimshoni, 2002

Shimshoni, Ilan; On Mobile Robot Localization From Landmark Bearings; IEEE Transactions on Robotics and Automation, Vol. 18, Nº 6, p.971-976, Dezembro de 2002.

SICK, 2003

SICK AG; NAV 200 Positioning System for Navigational Support – Technical Description; 2003.

SIEMENS, 2002a

Siemens Dematic Corp.; Model 8622, DT-20 Automatic Guided Tow Vehicle *(datasheet)*; 2002.

SIEMENS, 2002b

Siemens Dematic Corp.; Model 8635, DT-100 Automatic Guided Tow Vehicle *(datasheet)*; 2002.

Soares e Restivo, 1997

Soares, Manuel G.; Restivo, Francisco J.; Precisão Em Navegação Contínua Terrestre e Marítima com DGPS e GPSI; 3º Encontro Nacional do Colégio de Engenharia Electrotécnica, Porto (Exponor), 5 e 6 de Junho de 1997.

SPS-PS, 2001

Global Positioning System Standard Positioning Service Performance Standard; Assistant Secretary of Defense for Command, Control, Communications, and Intelligence, Outubro de 2001.

Stella *et al.*, 1995

Stella, E.; Altini, G; Lovergine, F. P. e Distante, A.; An Autonomous System for Indoor Structured Environment; Intelligent Autonomous Systems, U. Rembold *et al.* (Eds.), IOS Press, 1995.

Sutherland, 1994

Sutherland, Karen T.; The Stability of Geometric Inference in Location Determination; University of Utah Department of Computer Science Technical Report UUCS-94-021, Julho de 1994.

Sutherland e Thompson, 1993

Sutherland, K.T. e Thompson, W.B.; Inexact navigation; Proceedings of the IEEE International Conference on Robotics and Automation, Vol. 1, p. 1-7, 2 a 6 de Maio de 1993.

Sutherland e Thompson, 1994

Sutherland, Karen T. e Thompson, William B.; Localizing in Unstructured Environments: Dealing with the Errors; IEEE Transactions on Robotics and Automation, Vol. 10, Nº 6, p. 740-754, Dezembro de 1994.

Tanaka *et al.*, 2003

Tanaka, Kanji; Hasegawa, Tsutomu; Zha, Hongbin; Kondo, Eiji e Okada, Nobuhiro; Mobile Robot Localization with an Incomplete Map in Non-Stationary Environments; Proceedings of the 2003 IEEE International Conference on Robotics and Automation, p. 2848-2853, Taipei, Taiwan, 14 a 19 de Setembro de 2003.

Taylor, 1997

Taylor, John R.; An Introduction to Error Analysis – The Study of Uncertainties in Physical Measurements (2ª ed.); University Science Books, 1997.

Teunon, 1992

Teunon, I.; Principles of Active and Passive Systems; International Training on Elecronic Identification and Monitoring Within Animal Husbandry, COMETT II Course, 22-25 de Abril de 1992.

Thompson *et al.*, 1996

Thompson, William B.; Bennett, Bonnie H.; Sutherland, Karen T.; Geometric Reasoning for Map-Based Localization; University of Utah Department of Computer Science Technical Report UUCS-96-006, Maio de 1996.

TNEB, 1990

Robot; The New Encyclopaedia Britannica (15ª ed.) – Micropaedia, Ready-Reference; Vol. 10, p. 116, 1990.

Torres e Lima, 1996

Torres, João Agria; Lima, José Nuno; O Sistema Global de Posicionamento; Ingenium, 2ª Série, Nº 12, p. 69-76, Outubro de 1996.

Trimble, 2000

Trimble Navigation Limited; MS860 Rugged Dual-Antenna GPS Receiver for Precise Heading and Position *(datasheet)*; 2000.

Trimble, 2002

Trimble Navigation Limited; MS750 Dual Frequency RTK Receiver for Precise Dynamic Positioning *(datasheet)*; 2002.

Turennout e Honderd, 1992

Turennout, P. van; Honderd, G.; Navigation of a Mobile Robot; Tzafestas, S. G. (ed.), Robotic Systems – Advanced Techniques and Applications, p. 391-398; Kluwer Academic Publishers, 1992.

Venet *et al.*, 2002

Venet, T.; Capitaine, T.; Hamzaoui, M. e Fazzino, F.; One active beacon for an indoor absolute localization of a mobile vehicle; Proceedings of the 2002 IEEE International Conference on Robotics and Automation, p. 1-6, Washington, DC, Maio de 2002.

Wijk e Christensen, 2000

Wijk, Olle e Christensen, Henrik I.; Triangulation-Based Fusion of Sonar Data with Application in Robot Pose Tracking; IEEE Transactions on Robotics and Automation, Vol. 16, Nº 6, p. 740-752, Dezembro de 2000.

Yuen e MacDonald, 2002

Yuen, David C. K. e MacDonald, Bruce A.; Natural Landmark Based Localisation System Using Panoramic Images; Proceedings of the 2002 IEEE International Conference on Robotics and Automation, p. 915-920, Washington, DC, Maio de 2002.

Zhao, 1997

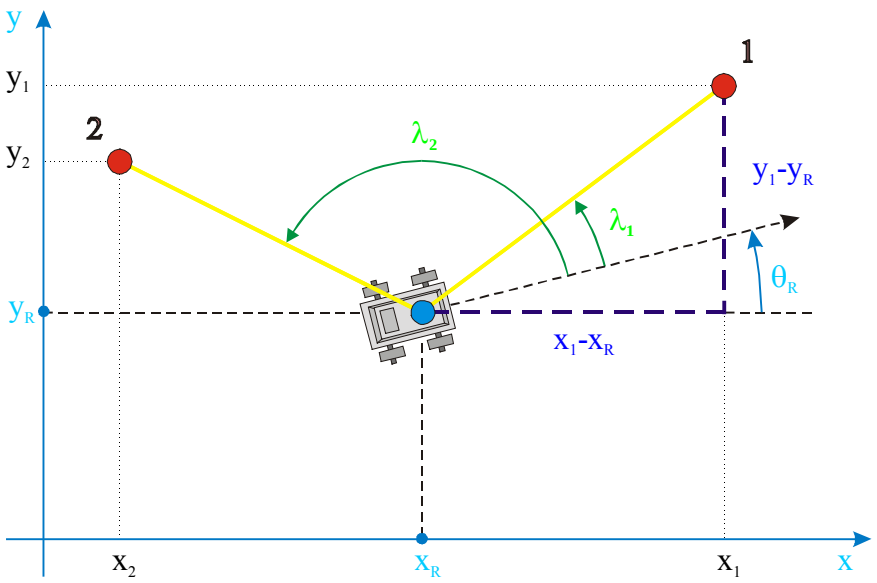
Zhao, Yilin; Vehicle Location and Navigation Systems; Artech House, 1997.

Anexos

A. Triangulação com Duas Balizas e Orientação Conhecida

Dadas duas balizas distinguíveis situadas em posições conhecidas do plano de navegação e medidos os ângulos λ_1 e λ_2 (Figura A.1) existentes entre um semi-eixo de referência fixo no robô e os segmentos de recta que unem o robô a cada baliza e também o ângulo θ_R existente entre esse eixo de referência e o semi-eixo positivo dos xx num dado instante, é possível determinar a posição do robô (coordenadas x_R e y_R) no referencial x0y definido no plano de navegação, sem recorrer a suposições sobre movimentos anteriores, utilizando as seguintes expressões:

$$\begin{cases} \dfrac{y_1 - y_R}{x_1 - x_R} = \operatorname{tg}(\lambda_1 + \theta_R) \\ \dfrac{y_2 - y_R}{x_2 - x_R} = \operatorname{tg}(\lambda_2 + \theta_R) \end{cases} \Rightarrow \begin{cases} x_R = \dfrac{y_1 - y_2 - x_1 \operatorname{tg}(\lambda_1 + \theta_R) + x_2 \operatorname{tg}(\lambda_2 + \theta_R)}{\operatorname{tg}(\lambda_2 + \theta_R) - \operatorname{tg}(\lambda_1 + \theta_R)} \\ y_R = y_1 - (x_1 - x_R)\operatorname{tg}(\lambda_1 + \theta_R) \end{cases} \quad \text{(A.1)}$$



λ_1 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **1**

λ_2 - Ângulo formado por um eixo de referência fixo no robô com o segmento de recta que une o robô à baliza **2**

x_1 , y_1 - Coordenadas da baliza **1**

x_2 , y_2 - Coordenadas da baliza **2**

θ_R - Orientação do robô

x_R , y_R - Coordenadas do robô no referencial x0y

Figura A.1: Grandezas em jogo na Triangulação com duas balizas e orientação conhecida.

A posição do robô não pode ser calculada nas seguintes circunstâncias:

- $\text{tg}(\lambda_1 + \theta_R) = \text{tg}(\lambda_2 + \theta_R)$ (as duas balizas e o robô são colineares);
- $x_1 = x_R$ $\quad (\lambda_1 + \theta_R = 90^\circ \text{ ou } \lambda_1 + \theta_R = 270^\circ)$; (A.2)
- $x_2 = x_R$ $\quad (\lambda_2 + \theta_R = 90^\circ \text{ ou } \lambda_2 + \theta_R = 270^\circ)$.

Como resultado das restrições indicadas, o robô não se pode localizar sobre as linhas representadas na Figura A.2.

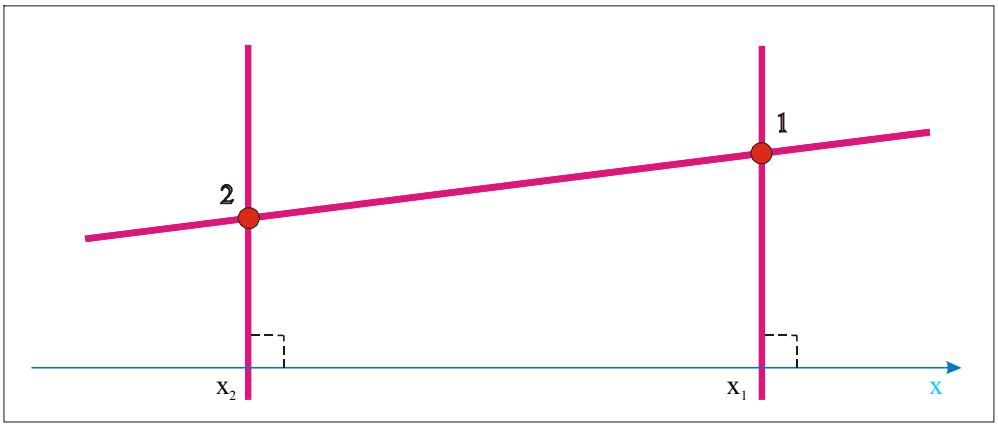


Figura A.2: Linhas sobre as quais o robô não se pode localizar.

B. Exemplo de Ambiguidade na Orientação Calculada com o Algoritmo Simples de Triangulação

$$\text{Situação A:} \begin{cases} x_R = 10 \\ y_R = 6 \\ \theta_R = 14{,}036^\circ \end{cases} \Rightarrow \begin{cases} \lambda_1 = 22{,}834^\circ \\ \lambda_2 = 139{,}399^\circ \\ \lambda_3 = 300{,}964^\circ \end{cases} \tag{B.1}$$

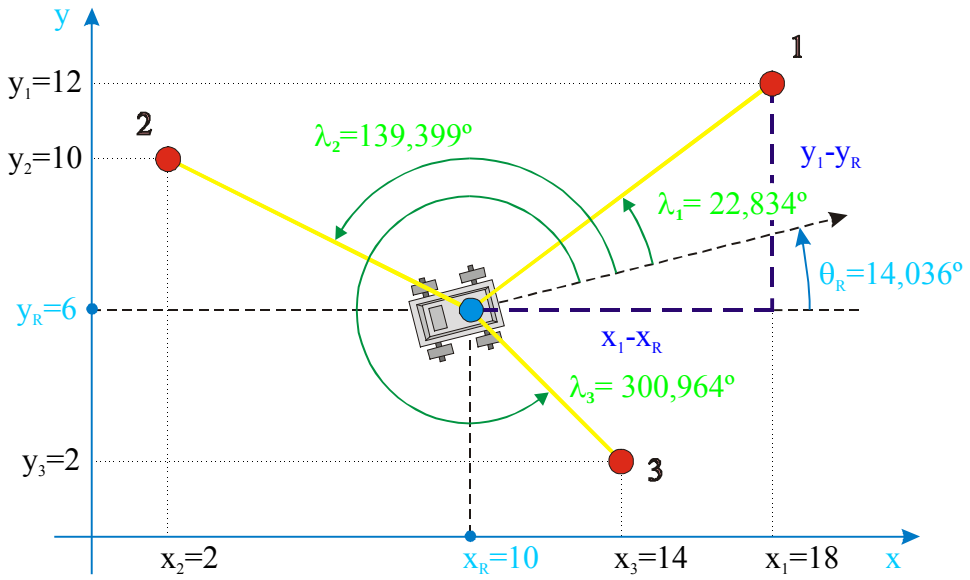


Figura B.1: Localização de um robô situado no ponto de coordenadas (10,6) do referencial x0y, uma orientação θ_R=14,036º

$$\begin{cases} \dfrac{12-y}{18-x} = \operatorname{tg}(22{,}834^\circ+\theta) = \dfrac{\operatorname{tg}(22{,}834^\circ)+\operatorname{tg}\theta}{1-\operatorname{tg}(22{,}834^\circ)\operatorname{tg}\theta} \\ \dfrac{10-y}{2-x} = \operatorname{tg}(139{,}399^\circ+\theta) = \dfrac{\operatorname{tg}(139{,}399^\circ)+\operatorname{tg}\theta}{1-\operatorname{tg}(139{,}399^\circ)\operatorname{tg}\theta} \\ \dfrac{2-y}{14-x} = \operatorname{tg}(300{,}964^\circ+\theta) = \dfrac{\operatorname{tg}(300{,}964^\circ)+\operatorname{tg}\theta}{1-\operatorname{tg}(300{,}964^\circ)\operatorname{tg}\theta} \end{cases} \tag{B.2}$$

$$\Rightarrow \begin{cases} \dfrac{12-y}{18-x} = \dfrac{0{,}42106+\operatorname{tg}\theta}{1-0{,}42106\operatorname{tg}\theta} \\ \dfrac{10-y}{2-x} = \dfrac{-0{,}85713+\operatorname{tg}\theta}{1+0{,}85713\operatorname{tg}\theta} \\ \dfrac{2-y}{14-x} = \dfrac{-1{,}6667+\operatorname{tg}\theta}{1+1{,}6667\operatorname{tg}\theta} \end{cases} \tag{B.3}$$

$$\underline{\text{Situação B}}:\ \begin{cases} x_R = 10 \\ y_R = 6 \\ \theta_R = 14{,}036^\circ + 180^\circ = 194{,}036^\circ \end{cases} \Rightarrow \begin{cases} \lambda_1 = 202{,}834^\circ \\ \lambda_2 = 319{,}399^\circ \\ \lambda_3 = 120{,}964^\circ \end{cases} \qquad \text{(B.4)}$$

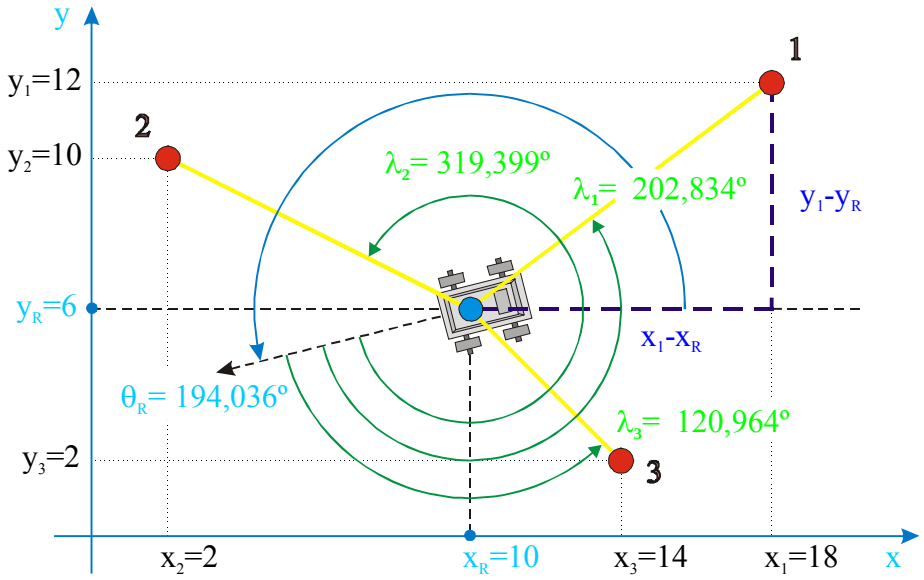


Figura B.2: Localização de um robô situado no ponto de coordenadas (10,6) do referencial x0y, uma orientação θ_R=194,036º

$$\begin{cases} \dfrac{12-y}{18-x} = \operatorname{tg}(202{,}834^\circ+\theta) = \dfrac{\operatorname{tg}(202{,}834^\circ)+\operatorname{tg}\theta}{1-\operatorname{tg}(202{,}834^\circ)\operatorname{tg}\theta} \\ \dfrac{10-y}{2-x} = \operatorname{tg}(319{,}399^\circ+\theta) = \dfrac{\operatorname{tg}(319{,}399^\circ)+\operatorname{tg}\theta}{1-\operatorname{tg}(319{,}399^\circ)\operatorname{tg}\theta} \\ \dfrac{2-y}{14-x} = \operatorname{tg}(120{,}964^\circ+\theta) = \dfrac{\operatorname{tg}(120{,}964^\circ)+\operatorname{tg}\theta}{1-\operatorname{tg}(120{,}964^\circ)\operatorname{tg}\theta} \end{cases} \qquad \text{(B.5)}$$

$$\Rightarrow \begin{cases} \dfrac{12-y}{18-x} = \dfrac{0{,}42106+\operatorname{tg}\theta}{1-0{,}42106\operatorname{tg}\theta} \\ \dfrac{10-y}{2-x} = \dfrac{-0{,}85713+\operatorname{tg}\theta}{1+0{,}85713\operatorname{tg}\theta} \\ \dfrac{2-y}{14-x} = \dfrac{-1{,}6667+\operatorname{tg}\theta}{1+1{,}6667\operatorname{tg}\theta} \end{cases} \qquad \text{(B.6)}$$

Apesar de a orientação do robô ser diferente nas situações A e B, o sistema (B.3) é idêntico ao sistema (B.6), por isso a solução obtida em ambos os casos será a mesma. Ao valor de tgθ obtido correspondem dois ângulos: θ_R e θ_R+180º.

C. Especificação do Algoritmo de Triangulação Baseado na Pesquisa Iterativa

O Algoritmo de Triangulação Baseado na Pesquisa Iterativa é descrito por Cohen e Koss (1992). A especificação apresentada no Capítulo 4 começa por considerar as três equações em x, y e θ (Figura C.1) do seguinte sistema, cuja resolução permite determinar x_R, y_R e θ_R (ou θ_R+180º):

$$\begin{cases} \dfrac{y_1 - y}{x_1 - x} = \operatorname{tg}(\lambda_1 + \theta) \Rightarrow & y = y_1 - (x_1 - x)\cdot \operatorname{tg}(\lambda_1 + \theta) \\ \dfrac{y_2 - y}{x_2 - x} = \operatorname{tg}(\lambda_2 + \theta) \Rightarrow & y = y_2 - (x_2 - x)\cdot \operatorname{tg}(\lambda_2 + \theta) \\ \dfrac{y_3 - y}{x_3 - x} = \operatorname{tg}(\lambda_3 + \theta) \Rightarrow & y = y_3 - (x_3 - x)\cdot \operatorname{tg}(\lambda_3 + \theta) \end{cases} \quad \text{(C.1)}$$

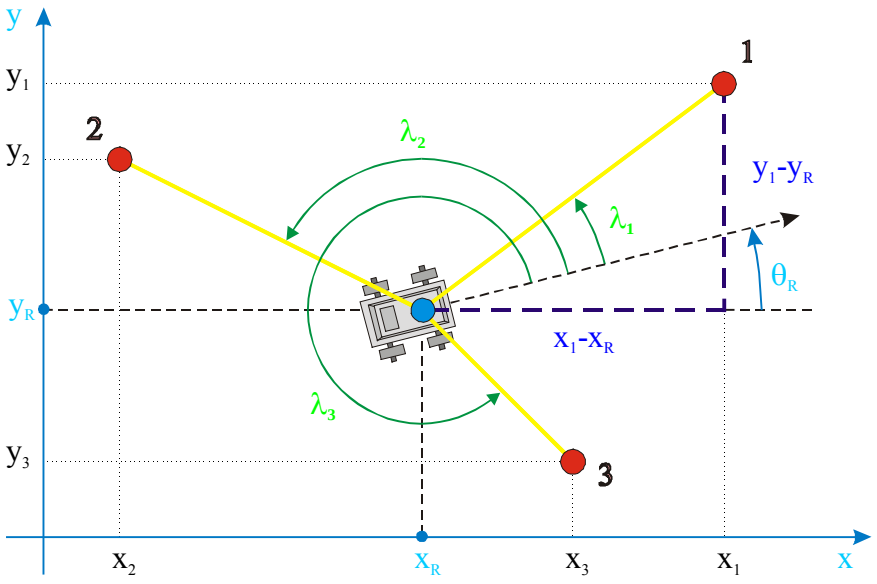


Figura C.1: Triangulação Baseada na Pesquisa Iterativa.

Seguidamente, consideram-se três soluções (A, B e C) para as coordenadas de posição do robô (estas soluções coincidem quando $\theta=\theta_R$):

Solução A (Utiliza as balizas **1** e **2**):

$$\begin{cases} y_1-(x_1-x_{RA})\cdot tg(\lambda_1+\theta)=y_2-(x_2-x_{RA})\cdot tg(\lambda_2+\theta) \\ y_{RA}=y_1-(x_1-x_{RA})\cdot tg(\lambda_1+\theta) \end{cases} \Rightarrow \begin{cases} x_{RA}=\dfrac{y_1-y_2-x_1\cdot tg(\lambda_1+\theta)+x_2\cdot tg(\lambda_2+\theta)}{tg(\lambda_2+\theta)-tg(\lambda_1+\theta)} \\ y_{RA}=y_1-(x_1-x_{RA})\cdot tg(\lambda_1+\theta) \end{cases} \quad (C.2)$$

Solução B (Utiliza as balizas **2** e **3**):

$$\begin{cases} y_2-(x_2-x_{RB})\cdot tg(\lambda_2+\theta)=y_3-(x_3-x_{RB})\cdot tg(\lambda_3+\theta) \\ y_{RB}=y_2-(x_2-x_{RB})\cdot tg(\lambda_2+\theta) \end{cases} \Rightarrow \begin{cases} x_{RB}=\dfrac{y_2-y_3-x_2\cdot tg(\lambda_2+\theta)+x_3\cdot tg(\lambda_3+\theta)}{tg(\lambda_3+\theta)-tg(\lambda_2+\theta)} \\ y_{RB}=y_2-(x_2-x_{RB})\cdot tg(\lambda_2+\theta) \end{cases} \quad (C.3)$$

Solução C (Utiliza as balizas **1** e **3**):

$$\begin{cases} y_1-(x_1-x_{RC})\cdot tg(\lambda_1+\theta)=y_3-(x_3-x_{RC})\cdot tg(\lambda_3+\theta) \\ y_{RC}=y_1-(x_1-x_{RC})\cdot tg(\lambda_1+\theta) \end{cases} \Rightarrow \begin{cases} x_{RC}=\dfrac{y_1-y_3-x_1\cdot tg(\lambda_1+\theta)+x_3\cdot tg(\lambda_3+\theta)}{tg(\lambda_3+\theta)-tg(\lambda_1+\theta)} \\ y_{RC}=y_1-(x_1-x_{RC})\cdot tg(\lambda_1+\theta) \end{cases} \quad (C.4)$$

Nas expressões anteriores, varia-se θ de -90º a +90º e calcula-se a área do triângulo formado pelas soluções A, B e C:

$$P_\Delta=\sqrt{(x_{RA}-x_{RB})^2+(y_{RA}-y_{RB})^2}+\sqrt{(x_{RB}-x_{RC})^2+(y_{RB}-y_{RC})^2}+\sqrt{(x_{RA}-x_{RC})^2+(y_{RA}-y_{RC})^2} \quad (C.5)$$

Os valores x_R e y_R são obtidos quando $\theta=\theta_R$ (ou $\theta=\theta_R+180º$), que produz o resultado

$$\begin{cases} x_{RA}=x_{RB}=x_{RC} \\ y_{RA}=y_{RB}=y_{RC} \end{cases} \Rightarrow P_\Delta=0 \quad (C.6)$$

Na prática, utiliza-se a solução aproximada correspondente ao valor de θ que produz o menor valor de P_Δ.

D. Especificação do Algoritmo de Triangulação Baseado no Método de Newton-Raphson

Este algoritmo é descrito por Cohen e Koss (1992). A especificação apresentada no Capítulo 4 começa por considerar as três equações em x, y e θ (Figura D.1) do seguinte sistema, cuja resolução permite determinar x_R, y_R e θ_R (ou θ_R+180º):

$$\begin{cases} \dfrac{y_1 - y}{x_1 - x} = \operatorname{tg}(\lambda_1 + \theta) \\ \dfrac{y_2 - y}{x_2 - x} = \operatorname{tg}(\lambda_2 + \theta) \\ \dfrac{y_3 - y}{x_3 - x} = \operatorname{tg}(\lambda_3 + \theta) \end{cases} \tag{D.1}$$

A partir deste sistema obtêm-se as seguintes funções:

$$\begin{cases} f_1(x, y, \theta) = \dfrac{y_1 - y}{x_1 - x} - \operatorname{tg}(\lambda_1 + \theta) = 0 \\ f_2(x, y, \theta) = \dfrac{y_2 - y}{x_2 - x} - \operatorname{tg}(\lambda_2 + \theta) = 0 \\ f_3(x, y, \theta) = \dfrac{y_3 - y}{x_3 - x} - \operatorname{tg}(\lambda_3 + \theta) = 0 \end{cases} \tag{D.2}$$

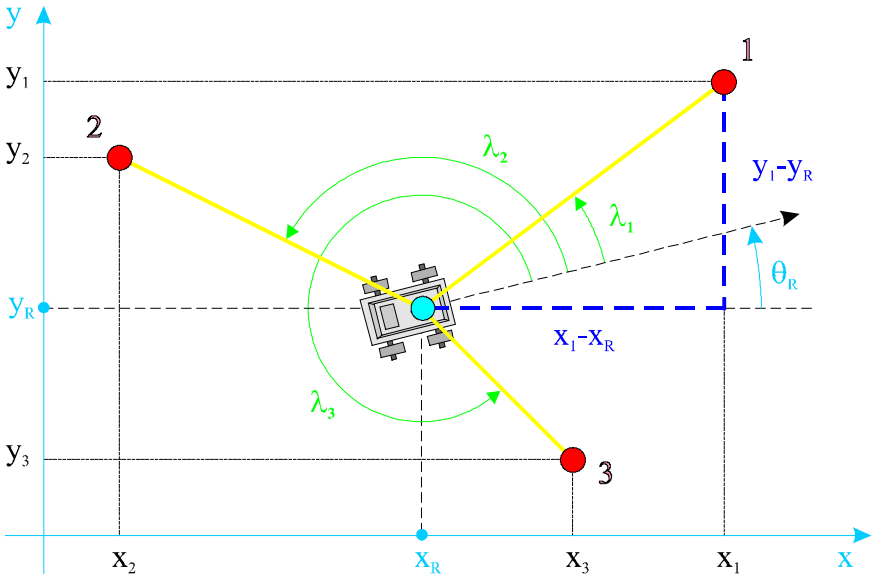


Figura D.1: Triangulação Baseada no Método de Newton-Raphson.

Seja o vector V a estimativa da solução no início de cada iteração:

$$V = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} \tag{D.3}$$

A estimativa da solução no final de cada iteração é dada por

$$\begin{bmatrix} x_i^{novo} \\ y_i^{novo} \\ \theta_i^{novo} \end{bmatrix} = V + \Delta V = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \theta \end{bmatrix} \tag{D.4}$$

em que

$$\Delta V = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \theta \end{bmatrix} = \begin{bmatrix} x_i^{novo} - x_i \\ y_i^{novo} - y_i \\ \theta_i^{novo} - \theta_i \end{bmatrix} \tag{D.5}$$

O desenvolvimento de $\mathbf{f_1}$, $\mathbf{f_2}$ e $\mathbf{f_3}$ nas respectivas séries de MacLaurin[1] resulta em

$$\begin{cases} f_1(V+\Delta V) = f_1(V) + \left(\dfrac{\partial f_1}{\partial x}\right)_V \cdot \Delta x + \left(\dfrac{\partial f_1}{\partial y}\right)_V \cdot \Delta y + \left(\dfrac{\partial f_1}{\partial \theta}\right)_V \cdot \Delta \theta \\ f_2(V+\Delta V) = f_2(V) + \left(\dfrac{\partial f_2}{\partial x}\right)_V \cdot \Delta x + \left(\dfrac{\partial f_2}{\partial y}\right)_V \cdot \Delta y + \left(\dfrac{\partial f_2}{\partial \theta}\right)_V \cdot \Delta \theta \\ f_3(V+\Delta V) = f_3(V) + \left(\dfrac{\partial f_3}{\partial x}\right)_V \cdot \Delta x + \left(\dfrac{\partial f_3}{\partial y}\right)_V \cdot \Delta y + \left(\dfrac{\partial f_3}{\partial \theta}\right)_V \cdot \Delta \theta \end{cases} \tag{D.6}$$

Por definição,

$$\begin{cases} f_1(V+\Delta V) = 0 \\ f_2(V+\Delta V) = 0 \\ f_3(V+\Delta V) = 0 \end{cases} \tag{D.7}$$

[1] O desenvolvimento em *série de Taylor* de uma função f(x) na vizinhança do ponto x=a é o seguinte:

$$f(x) = f(a) + \frac{(x-a)}{1} \cdot f'(a) + \frac{(x-a)^2}{2 \cdot 1} \cdot f''(a) + \ldots + \frac{(x-a)^n}{n!} \cdot f^n(a) + \ldots$$

Se a=0, obtém-se um caso particular da série de Taylor a que se chama *série de MacLaurin* (Piskounov, 1997).

Resultando, na forma matricial (as incógnitas são Δx, Δy e $\Delta\theta$),

$$\begin{bmatrix} \left(\frac{\partial f_1}{\partial x}\right)_V & \left(\frac{\partial f_1}{\partial y}\right)_V & \left(\frac{\partial f_1}{\partial \theta}\right)_V \\ \left(\frac{\partial f_2}{\partial x}\right)_V & \left(\frac{\partial f_2}{\partial y}\right)_V & \left(\frac{\partial f_2}{\partial \theta}\right)_V \\ \left(\frac{\partial f_3}{\partial x}\right)_V & \left(\frac{\partial f_3}{\partial y}\right)_V & \left(\frac{\partial f_3}{\partial \theta}\right)_V \end{bmatrix} \times \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} -f_1(V) \\ -f_2(V) \\ -f_3(V) \end{bmatrix} \tag{D.8}$$

ou seja,

$$\begin{bmatrix} \frac{y_1 - y_i}{(x_1 - x_i)^2} & \frac{1}{x_1 - x_i} & -1 - tg^2(\lambda_1 + \theta_i) \\ \frac{y_2 - y_i}{(x_2 - x_i)^2} & \frac{1}{x_2 - x_i} & -1 - tg^2(\lambda_2 + \theta_i) \\ \frac{y_3 - y_i}{(x_3 - x_i)^2} & \frac{1}{x_3 - x_i} & -1 - tg^2(\lambda_3 + \theta_i) \end{bmatrix} \times \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} -\frac{y_1 - y_i}{x_1 - x_i} + tg(\lambda_1 + \theta_i) \\ -\frac{y_2 - y_i}{x_2 - x_i} + tg(\lambda_2 + \theta_i) \\ -\frac{y_3 - y_i}{x_3 - x_i} + tg(\lambda_3 + \theta_i) \end{bmatrix} \tag{D.9}$$

Uma vez resolvido o anterior sistema de equações, a nova estimativa da solução é dada por

$$\begin{bmatrix} x_i^{novo} \\ y_i^{novo} \\ \theta_i^{novo} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix} \tag{D.10}$$

Por fim, faz-se

$$V = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} \equiv \begin{bmatrix} x_i^{novo} \\ y_i^{novo} \\ \theta_i^{novo} \end{bmatrix} \tag{D.11}$$

e o processo recomeça com a determinação dos novos valores de Δx, Δy e $\Delta\theta$, conducentes à obtenção de uma nova estimativa ainda mais aproximada da solução.

E. Dedução de Expressões Utilizadas na Localização por Trilateração

Para determinar $\mathbf{x_R}$ e $\mathbf{y_R}$ (Figura E.1) pode recorrer-se ao seguinte sistema de equações não lineares em **x** e **y**:

$$\begin{cases} L_1^{\,2} = (x - x_1)^2 + (y - y_1)^2 \\ L_2^{\,2} = (x - x_2)^2 + (y - y_2)^2 \\ L_3^{\,2} = (x - x_3)^2 + (y - y_3)^2 \end{cases} \tag{E.1}$$

Subtraindo a segunda e a terceira equação à primeira, obtém-se

$$\begin{cases} L_1^{\,2} - L_2^{\,2} = (x - x_1)^2 + (y - y_1)^2 - (x - x_2)^2 - (y - y_2)^2 \\ L_1^{\,2} - L_3^{\,2} = (x - x_1)^2 + (y - y_1)^2 - (x - x_3)^2 - (y - y_3)^2 \end{cases} \tag{E.2}$$

que se pode rescrever

$$\begin{cases} L_1^{\,2} - L_2^{\,2} - x_1^{\,2} + x_2^{\,2} - y_1^{\,2} + y_2^{\,2} = 2(x_2 - x_1)\cdot x + 2(y_2 - x_1)\cdot y \\ L_1^{\,2} - L_3^{\,2} - x_1^{\,2} + x_3^{\,2} - y_1^{\,2} + y_3^{\,2} = 2(x_3 - x_1)\cdot x + 2(y_3 - x_1)\cdot y \end{cases} \tag{E.3}$$

ou ainda, na forma matricial,

$$\begin{bmatrix} 2(x_2 - x_1) & 2(y_2 - x_1) \\ 2(x_3 - x_1) & 2(y_3 - x_1) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} L_1^{\,2} - L_2^{\,2} - x_1^{\,2} + x_2^{\,2} - y_1^{\,2} + y_2^{\,2} \\ L_1^{\,2} - L_3^{\,2} - x_1^{\,2} + x_3^{\,2} - y_1^{\,2} + y_3^{\,2} \end{bmatrix} \tag{E.4}$$

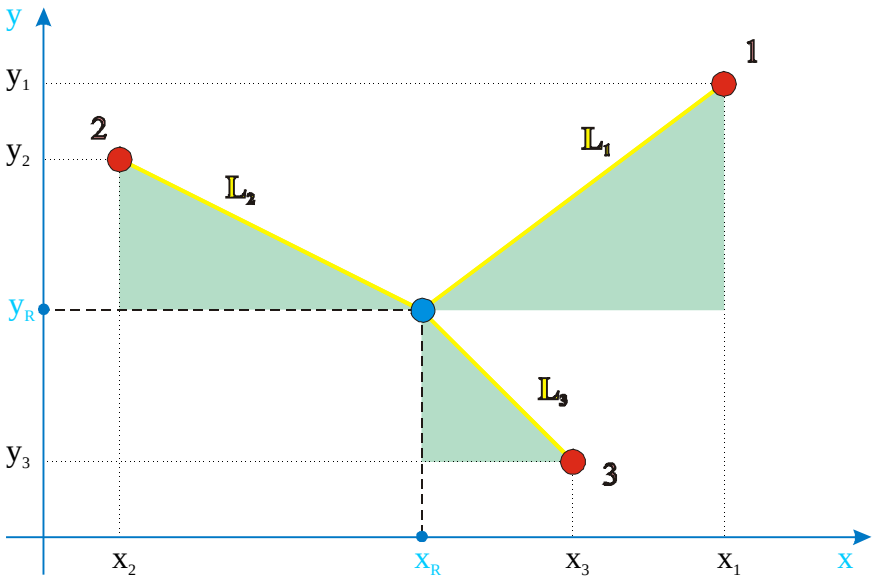


Figura E.1: Localização por Trilateração.

F. Dedução do Sistema de Equações Usado no Algoritmo de Triangulação com Intersecção de Três Circunferências para Determinar x_R e y_R

O Algoritmo de Triangulação com Intersecção de Três Circunferências, utilizado por Fuentes *et al.* (1995), inclui a resolução de um sistema de duas equações lineares em **x** e **y** para calcular $\mathbf{x_R}$ e $\mathbf{y_R}$ (Figura F.1) – coordenadas do robô no referencial x0y – a partir das coordenadas das balizas e dos raios das circunferências definidas pelo robô e por cada par de balizas.

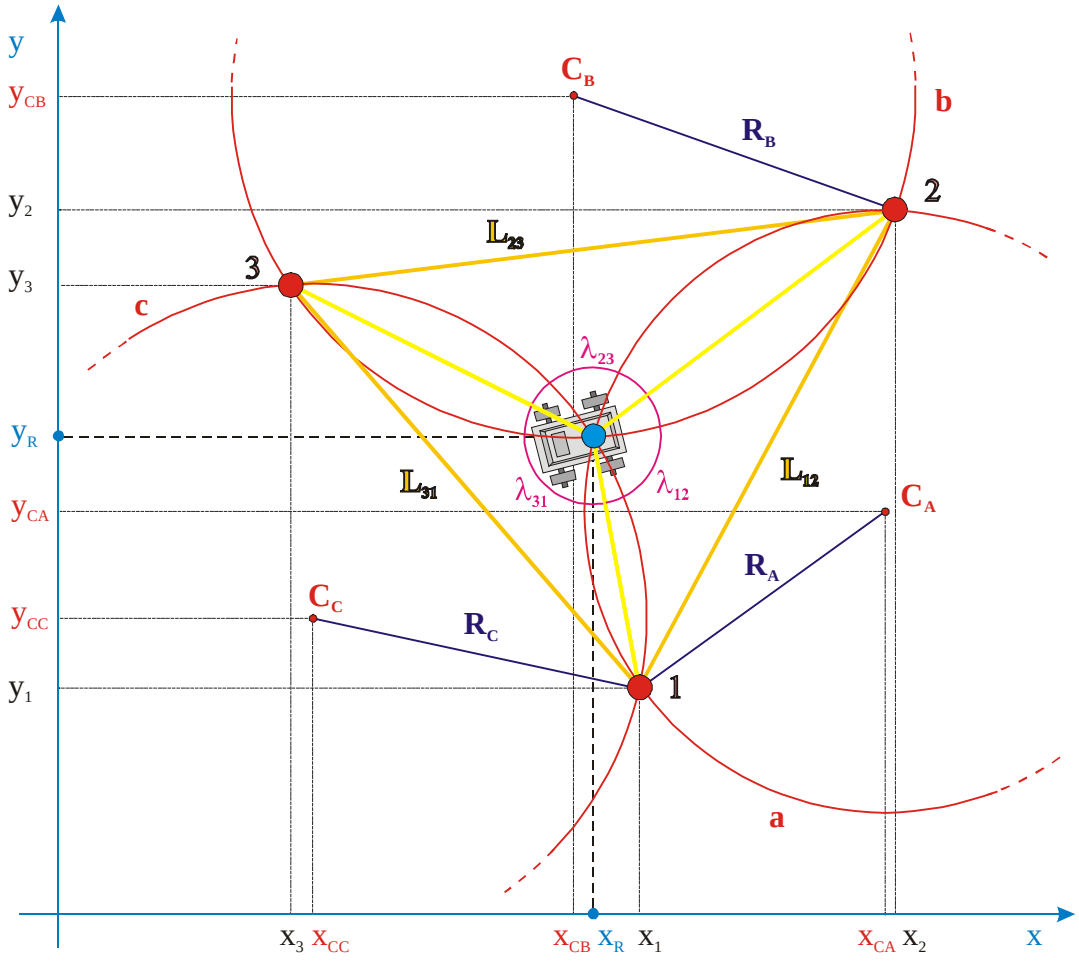


Figura F.1: Grandezas em jogo no Algoritmo de Triangulação com Intersecção de Três Circunferências

O ponto de partida da dedução apresentada por Fuentes *et al*, (1995) é o seguinte sistema de equações não lineares em **x** e **y**, que tem por solução $\mathbf{x_R}$ e $\mathbf{y_R}$:

$$\begin{cases} (x-x_{CA})^2+(y-y_{CA})^2=R_A{}^2 \\ (x-x_{CB})^2+(y-y_{CB})^2=R_B{}^2 \\ (x-x_{CC})^2+(y-y_{CC})^2=R_C{}^2 \end{cases} \Rightarrow \begin{cases} (1) \quad x^2+x_{CA}{}^2-2\cdot x\cdot x_{CA}+y^2+y_{CA}{}^2-2\cdot y\cdot y_{CA}=R_A{}^2 \\ (2) \quad x^2+x_{CB}{}^2-2\cdot x\cdot x_{CB}+y^2+y_{CB}{}^2-2\cdot y\cdot y_{CB}=R_B{}^2 \\ (3) \quad x^2+x_{CC}{}^2-2\cdot x\cdot x_{CC}+y^2+y_{CC}{}^2-2\cdot y\cdot y_{CC}=R_C{}^2 \end{cases} \tag{F.1}$$

Subtraindo a segunda equação à primeira e a terceira à segunda, obtém-se

$$\begin{cases} (1-2) \quad x_{CA}{}^2-x_{CB}{}^2-2\cdot x\cdot(x_{CA}-x_{CB})+y_{CA}{}^2-y_{CB}{}^2-2\cdot y\cdot(y_{CA}-y_{CB})=R_A{}^2-R_B{}^2 \\ (2-3) \quad x_{CB}{}^2-x_{CC}{}^2-2\cdot x\cdot(x_{CB}-x_{CC})+y_{CB}{}^2-y_{CC}{}^2-2\cdot y\cdot(y_{CB}-y_{CC})=R_B{}^2-R_C{}^2 \end{cases} \tag{F.2}$$

que se pode rescrever

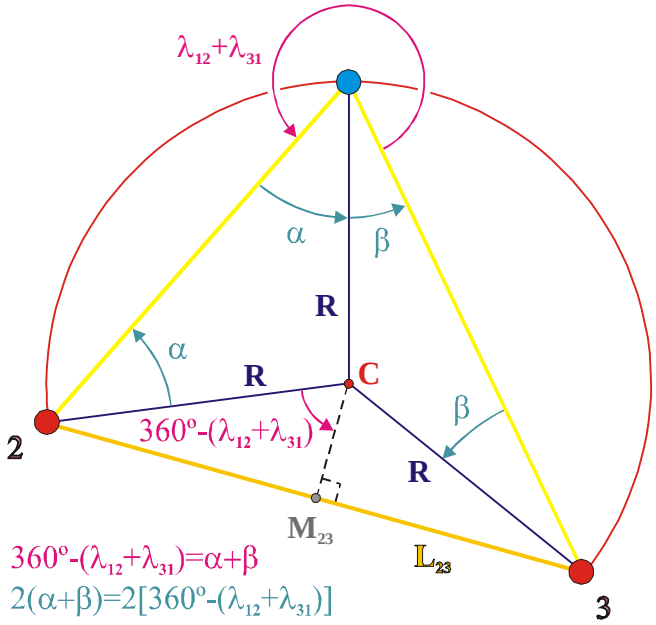
$$\begin{cases} 2\cdot x\cdot(x_{CA}-x_{CB})+2\cdot y\cdot(y_{CA}-y_{CB})=(x_{CA}{}^2-x_{CB}{}^2)+(y_{CA}{}^2-y_{CB}{}^2)-(R_A{}^2-R_B{}^2) \\ 2\cdot x\cdot(x_{CB}-x_{CC})+2\cdot y\cdot(y_{CB}-y_{CC})=(x_{CB}{}^2-x_{CC}{}^2)+(y_{CB}{}^2-y_{CC}{}^2)-(R_B{}^2-R_C{}^2) \end{cases} \tag{F.3}$$

ou ainda, na forma matricial,

$$\begin{bmatrix} 2\cdot(x_{CA}-x_{CB}) & 2\cdot(y_{CA}-y_{CB}) \\ 2\cdot(x_{CB}-x_{CC}) & 2\cdot(y_{CB}-y_{CC}) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} (x_{CA}{}^2-x_{CB}{}^2)+(y_{CA}{}^2-y_{CB}{}^2)-(R_A{}^2-R_B{}^2) \\ (x_{CB}{}^2-x_{CC}{}^2)+(y_{CB}{}^2-y_{CC}{}^2)-(R_B{}^2-R_C{}^2) \end{bmatrix} \tag{F.4}$$

G. Dedução das Expressões Utilizadas no Algoritmo da Figura 5.52 Para Calcular R e L_{CM23}

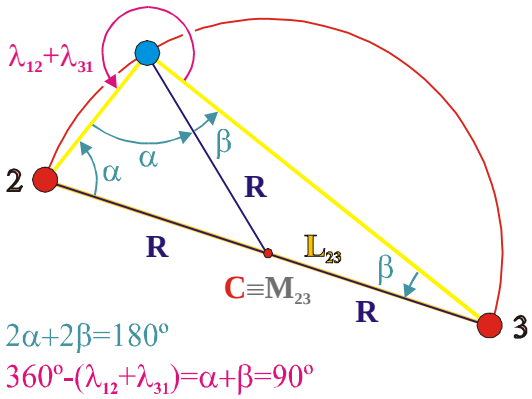
$\lambda_{12}+\lambda_{31} > 270^o$



$$\frac{L_{23}}{2} = R \cdot \text{sen}\left[360^o - (\lambda_{12}+\lambda_{31})\right] \Rightarrow R = -\frac{L_{23}}{2 \cdot \text{sen}(\lambda_{12}+\lambda_{31})} = \left|\frac{L_{23}}{2 \cdot \text{sen}(\lambda_{12}+\lambda_{31})}\right| \quad \text{(G.1)}$$

$$L_{CM23} = \frac{\frac{L_{23}}{2}}{\text{tg}\left[360^o - (\lambda_{12}+\lambda_{31})\right]} = -\frac{L_{23}}{2\,\text{tg}(\lambda_{12}+\lambda_{31})} > 0 \quad \text{(G.2)}$$

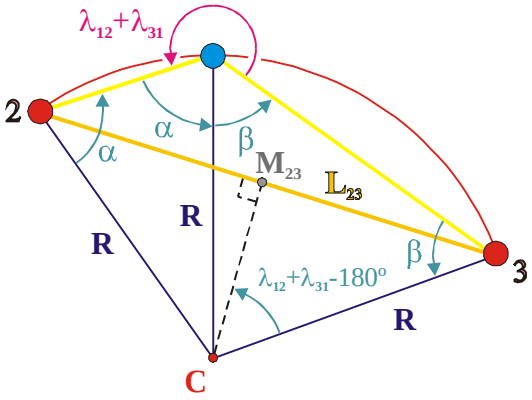
$\lambda_{12}+\lambda_{31} = 270^o$



$$R = \frac{L_{23}}{2} = -\frac{L_{23}}{2 \cdot \text{sen}\,270^o} = -\frac{L_{23}}{2 \cdot \text{sen}(\lambda_{12}+\lambda_{31})} = \left|\frac{L_{23}}{2 \cdot \text{sen}(\lambda_{12}+\lambda_{31})}\right| \quad \text{(G.3)}$$

$$L_{CM23} = -\frac{L_{23}}{2\,\text{tg}(\lambda_{12}+\lambda_{31})} = -\frac{L_{23}}{2\,\text{tg}(270^o)} = 0 \quad \text{(G.4)}$$

$180^o < \lambda_{12}+\lambda_{31} < 270^o$



$360^o-(\lambda_{12}+\lambda_{31})=\alpha+\beta$

$360^o-2(\alpha+\beta)=2(\lambda_{12}+\lambda_{31})-360^o$

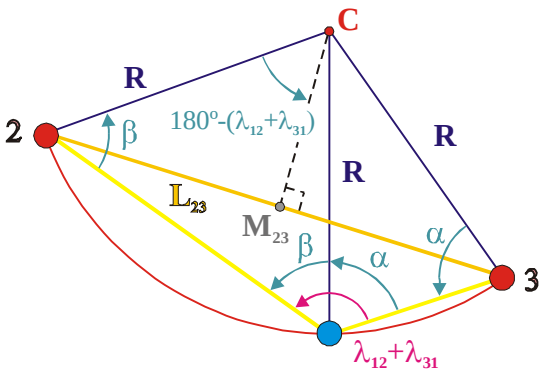
$$\frac{L_{23}}{2} = R\cdot\text{sen}(\lambda_{12}+\lambda_{31}-180^o) = -R\cdot\text{sen}(\lambda_{12}+\lambda_{31}) \Rightarrow R = -\frac{L_{23}}{2\cdot\text{sen}(\lambda_{12}+\lambda_{31})} = \left|\frac{L_{23}}{2\cdot\text{sen}(\lambda_{12}+\lambda_{31})}\right| \quad \text{(G.5)}$$

$$L_{CM23} = -\frac{\frac{L_{23}}{2}}{\text{tg}(\lambda_{12}+\lambda_{31}-180^o)} = -\frac{L_{23}}{2\,\text{tg}(\lambda_{12}+\lambda_{31})} < 0 \quad [1] \quad \text{(G.6)}$$

$90^o < \lambda_{12}+\lambda_{31} < 180^o$

$\lambda_{12}+\lambda_{31}=\alpha+\beta$

$360^o-2(\alpha+\beta)=360^o-2(\lambda_{12}+\lambda_{31})$

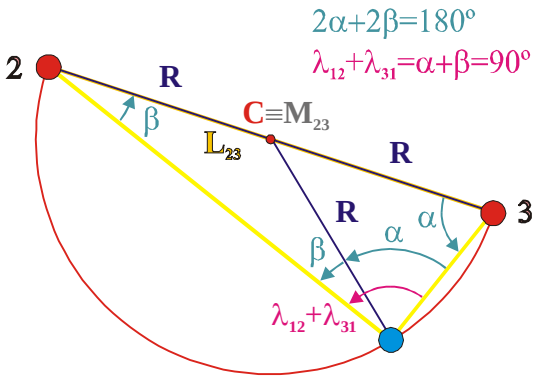


$$\frac{L_{23}}{2} = R\cdot\text{sen}[180^o-(\lambda_{12}+\lambda_{31})] = R\cdot\text{sen}(\lambda_{12}+\lambda_{31}) \Rightarrow R = \frac{L_{23}}{2\cdot\text{sen}(\lambda_{12}+\lambda_{31})} = \left|\frac{L_{23}}{2\cdot\text{sen}(\lambda_{12}+\lambda_{31})}\right| \quad \text{(G.7)}$$

$$L_{CM23} = \frac{\frac{L_{23}}{2}}{\text{tg}[180^o-(\lambda_{12}+\lambda_{31})]} = -\frac{L_{23}}{2\,\text{tg}(\lambda_{12}+\lambda_{31})} > 0 \quad \text{(G.8)}$$

[1] Coloca-se o sinal "–" nesta expressão pois é necessário que $L_{CM23}<0$ quando $180^o<\lambda_{12}+\lambda_{31}<270^o$, para que o algoritmo funcione correctamente.

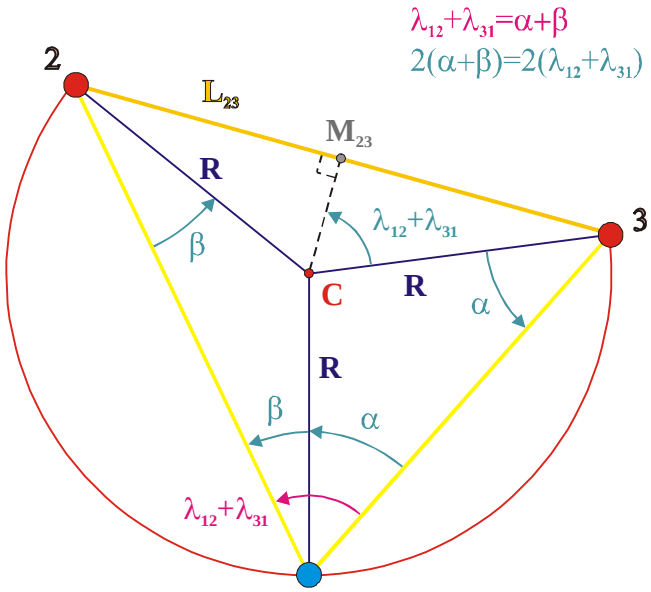
$\lambda_{12}+\lambda_{31} = 90^o$



$$R = \frac{L_{23}}{2} = \frac{L_{23}}{2\cdot \text{sen}\,90^o} = \frac{L_{23}}{2\cdot \text{sen}(\lambda_{12}+\lambda_{31})} = \left|\frac{L_{23}}{2\cdot \text{sen}(\lambda_{12}+\lambda_{31})}\right| \qquad \text{(G.9)}$$

$$L_{CM23} = -\frac{L_{23}}{2\,\text{tg}(\lambda_{12}+\lambda_{31})} = -\frac{L_{23}}{2\,\text{tg}(90^o)} = 0 \qquad \text{(G.10)}$$

$\lambda_{12}+\lambda_{31} < 90^o$



$$\frac{L_{23}}{2} = R\cdot \text{sen}(\lambda_{12}+\lambda_{31}) \;\Rightarrow\; R = \frac{L_{23}}{2\cdot \text{sen}(\lambda_{12}+\lambda_{31})} = \left|\frac{L_{23}}{2\cdot \text{sen}(\lambda_{12}+\lambda_{31})}\right| \qquad \text{(G.11)}$$

$$L_{CM23} = -\frac{\frac{L_{23}}{2}}{\text{tg}(\lambda_{12}+\lambda_{31})} = -\frac{L_{23}}{2\,\text{tg}(\lambda_{12}+\lambda_{31})} < 0 \;^{2} \qquad \text{(G.12)}$$

[2] Coloca-se o sinal "–" nesta expressão pois é necessário que $L_{CM23}<0$ quando $\lambda_{12}+\lambda_{31}<90^o$, para que o algoritmo funcione correctamente.

H. Caracterização da Elipse Φ

Neste anexo caracteriza-se a elipse Φ utilizada no ponto 5.6.5.

A equação cartesiana de uma cónica[1] no referencial ortonormado $\lambda_{12}0\lambda_{31}$ é a seguinte[2]:

$$A\lambda_{12}^2 + 2H\lambda_{12}\lambda_{31} + B\lambda_{31}^2 + 2D\lambda_{12} + 2F\lambda_{31} + G = 0 \tag{H.1}$$

O centro de simetria da cónica é o ponto com as seguintes coordenadas:

$$\mathbf{P0}\left(\frac{BD-HF}{H^2-AB}, \frac{AF-HD}{H^2-AB}\right), \quad H^2 - AB \neq 0 \tag{H.2}$$

A cónica é uma elipse se

$$H^2 - AB < 0 \tag{H.3}$$

As rectas que contêm os eixos da elipse possuem a equação

$$(aA+bH)\lambda_{12} + (aH+bB)\lambda_{31} + aD + bF = 0 \ ^3 \tag{H.4}$$

se, e só se, existir um ω tal que

$$\begin{cases} a(A-\omega) + bH = 0 \\ aH + b(B-\omega) = 0 \end{cases} \tag{H.5}$$

sistema este que possui uma solução não nula se, e só se,

$$\begin{vmatrix} A-\omega & H \\ H & B-\omega \end{vmatrix} = (A-\omega)(B-\omega) - H^2 = 0 \tag{H.6}$$

Para cada valor de ω que satisfaça a equação (H.6), resolve-se o sistema de equações (H.5) e substitui-se as soluções obtidas na equação (H.4). A intersecção de cada recta com a elipse permite determinar os seus vértices e, a partir destes, os comprimentos dos seus eixos (Novais, 1981).

[1] As expressões (H.1) a (H.6) são referidas por Novais (1981).

[2] A, H, B, D, F e G são constantes e A, H e B são não simultaneamente nulos.

[3] a e b são constantes e não simultaneamente nulos.

No caso particular em que

$$\begin{cases} A = \dfrac{2}{3m^2\sigma_\lambda^2} \\ B = \dfrac{2}{3m^2\sigma_\lambda^2} \\ H = \dfrac{1}{3m^2\sigma_\lambda^2} \\ D = -\dfrac{2\lambda_{12m} + \lambda_{31m}}{3m^2\sigma_\lambda^2} \\ F = -\dfrac{\lambda_{12m} + 2\lambda_{31m}}{3m^2\sigma_\lambda^2} \\ G = -\dfrac{2}{3m^2\sigma_\lambda^2} \cdot \left(\lambda_{12m}^2 + \lambda_{12m} \cdot \lambda_{31m} + \lambda_{31m}^2\right) - 1 \end{cases} \tag{H.7}$$

com m, σ_λ, λ_{12m} e λ_{31m} constantes, obtém-se a expressão seguinte, correspondente à equação de uma elipse Φ (Figura H.1) cujo centro de simetria é o ponto de coordenadas λ_{12m} e λ_{31m}:

$$\frac{2}{3m^2\sigma_\lambda^2} \cdot \left[(\lambda_{12} - \lambda_{12m})^2 + (\lambda_{12} - \lambda_{12m})(\lambda_{31} - \lambda_{31m}) + (\lambda_{31} - \lambda_{31m})^2\right] = 1 \tag{H.8}$$

Para facilitar o cálculo das coordenadas dos vértices desta elipse procede-se à seguinte mudança de variáveis[4]:

$$\begin{cases} \Lambda_{12} = \lambda_{12} - \lambda_{12m} \\ \Lambda_{31} = \lambda_{31} - \lambda_{31m} \end{cases} \Rightarrow \begin{cases} \lambda_{12} = \Lambda_{12} + \lambda_{12m} \\ \lambda_{31} = \Lambda_{31} + \lambda_{31m} \end{cases} \tag{H.9}$$

A equação da elipse Φ no referencial $\Lambda_{12}0'\Lambda_{31}$ é a seguinte

$$\frac{2}{3m^2\sigma_\lambda^2} \cdot \left[\Lambda_{12}{}^2 + \Lambda_{12}\Lambda_{31} + \Lambda_{31}{}^2\right] = 1 \tag{H.10}$$

Neste caso teremos

$$\begin{cases} A = B = \dfrac{2}{3m^2\sigma_\lambda^2} \\ H = \dfrac{1}{3m^2\sigma_\lambda^2} \\ D = F = 0 \\ G = -1 \end{cases} \tag{H.11}$$

[4] A orientação da elipse e os comprimentos dos seus eixos não mudam se os eixos coordenados sofrerem uma translação.

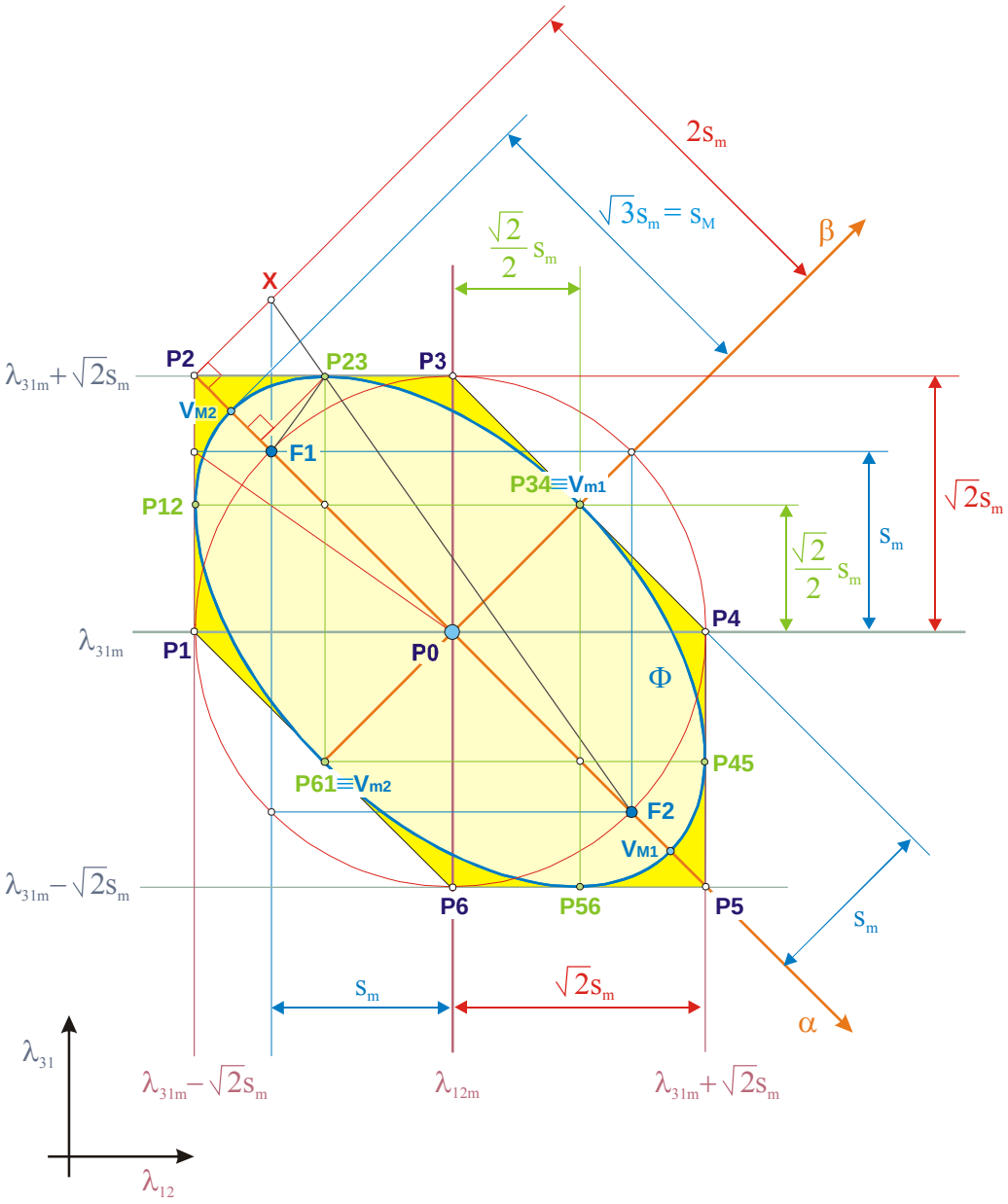


Figura H.1: A elipse Φ situa-se no referencial $\lambda_{12}0\lambda_{31}$ e está centrada no ponto de coordenadas λ_{12m} e λ_{31m}.

Substituindo em (H.6) os valores referidos em (H.11) obtém-se:

$$\begin{vmatrix} \frac{2}{3m^2\sigma_\lambda^2}-\omega & \frac{1}{3m^2\sigma_\lambda^2} \\ \frac{1}{3m^2\sigma_\lambda^2} & \frac{2}{3m^2\sigma_\lambda^2}-\omega \end{vmatrix} = \left(\frac{2}{3m^2\sigma_\lambda^2}-\omega\right)^2 - \left(\frac{1}{3m^2\sigma_\lambda^2}\right)^2 = 0 \tag{H.12}$$

$$\Rightarrow \quad \omega = \frac{1}{3m^2\sigma_\lambda^2} \quad \vee \quad \omega = \frac{1}{m^2\sigma_\lambda^2}$$

De acordo com o anteriormente exposto, para obter as coordenadas dos vértices $\mathbf{V_{m1}}$ e $\mathbf{V_{m2}}$ da elipse Φ, efectuam-se os seguintes cálculos:

$$\begin{cases} \omega = \dfrac{1}{3m^2\sigma_\lambda^2} \\ a(A-\omega)+bH=0 \\ aH+b(B-\omega)=0 \end{cases}$$

$$\Rightarrow \quad \begin{cases} a\cdot\left(\dfrac{2}{3m^2\sigma_\lambda^2}-\dfrac{1}{3m^2\sigma_\lambda^2}\right)+b\cdot\dfrac{1}{3m^2\sigma_\lambda^2}=0 \\ a\cdot\dfrac{1}{3m^2\sigma_\lambda^2}+b\cdot\left(\dfrac{2}{3\sigma_\lambda^2}-\dfrac{1}{3m^2\sigma_\lambda^2}\right)=0 \end{cases}$$

$$\Rightarrow \quad a=-b$$

$$\begin{cases} a=-b \\ (aA+bH)\Lambda_{12}+(aH+bB)\Lambda_{31}=0 \end{cases}$$

$$\Rightarrow \quad \left(a\frac{2}{3m^2\sigma_\lambda^2}-a\frac{1}{3m^2\sigma_\lambda^2}\right)\Lambda_{12}+\left(a\frac{1}{3m^2\sigma_\lambda^2}-a\frac{2}{3m^2\sigma_\lambda^2}\right)\Lambda_{31}=0 \qquad \text{(H.13)}$$

$$\Rightarrow \quad \Lambda_{12}=\Lambda_{31}$$

$$\begin{cases} \Lambda_{12}=\Lambda_{31} \\ A\Lambda_{12}^2+2H\Lambda_{12}\Lambda_{31}+B\Lambda_{31}^2+G=0 \end{cases}$$

$$\Rightarrow \frac{2}{3m^2\sigma_\lambda^2}\left(\Lambda_{12}^2+\Lambda_{12}^2+\Lambda_{12}^2\right)-1=0$$

$$\Rightarrow \Lambda_{12}^2=\frac{m^2\sigma_\lambda^2}{2}$$

$$\Rightarrow \quad \begin{cases} \Lambda_{12}=\pm\dfrac{m\sigma_\lambda}{\sqrt{2}} \\ \Lambda_{31}=\pm\dfrac{m\sigma_\lambda}{\sqrt{2}} \end{cases}$$

Do último resultado conclui-se que dois dos vértices da elipse Φ, que passam a chamar-se $\mathbf{V_{m1}}$ e $\mathbf{V_{m2}}$, possuem as seguintes coordenadas no referencial $\Lambda_{12}0'\Lambda_{31}$:

$$\mathbf{V_{m1}}\left(\frac{\sqrt{2}}{2}m\sigma_\lambda,\ \frac{\sqrt{2}}{2}m\sigma_\lambda\right)$$

$$\mathbf{V_{m2}}\left(-\frac{\sqrt{2}}{2}m\sigma_\lambda,\ -\frac{\sqrt{2}}{2}m\sigma_\lambda\right) \qquad \text{(H.14)}$$

De acordo com (H.9), as coordenadas destes vértices no referencial $\lambda_{12}0\lambda_{31}$ são as seguintes (Figura H.1):

$$\begin{aligned}
&\mathbf{V_{m1}} \left(\lambda_{12m} + \frac{\sqrt{2}}{2} m\sigma_\lambda, \quad \lambda_{31m} + \frac{\sqrt{2}}{2} m\sigma_\lambda \right) \\
&\mathbf{V_{m2}} \left(\lambda_{12m} - \frac{\sqrt{2}}{2} m\sigma_\lambda, \quad \lambda_{31m} - \frac{\sqrt{2}}{2} m\sigma_\lambda \right)
\end{aligned} \tag{H.15}$$

Para obter as coordenadas dos vértices $\mathbf{V_{M1}}$ e $\mathbf{V_{M2}}$ da elipse Φ, efectuam-se os seguintes cálculos:

$$\begin{cases} \omega = \dfrac{1}{m^2\sigma_\lambda^2} \\ a(A-\omega) + bH = 0 \\ aH + b(B-\omega) = 0 \end{cases}$$

$$\Rightarrow \quad \begin{cases} a \cdot \left(\dfrac{2}{3m^2\sigma_\lambda^2} - \dfrac{1}{m^2\sigma_\lambda^2} \right) + b \cdot \dfrac{1}{3m^2\sigma_\lambda^2} = 0 \\ a \cdot \dfrac{1}{3m^2\sigma_\lambda^2} + b \cdot \left(\dfrac{2}{3m^2\sigma_\lambda^2} - \dfrac{1}{m^2\sigma_\lambda^2} \right) = 0 \end{cases}$$

$$\Rightarrow \quad a = b$$

$$\begin{cases} a = b \\ (aA + bH)\Lambda_{12} + (aH + bB)\Lambda_{31} = 0 \end{cases}$$

$$\Rightarrow \quad \left(a\frac{2}{3m^2\sigma_\lambda^2} + a\frac{1}{3m^2\sigma_\lambda^2} \right)\Lambda_{12} + \left(a\frac{1}{3m^2\sigma_\lambda^2} + a\frac{2}{3m^2\sigma_\lambda^2} \right)\Lambda_{31} = 0 \tag{H.16}$$

$$\Rightarrow \quad \Lambda_{12} = -\Lambda_{31}$$

$$\begin{cases} \Lambda_{12} = -\Lambda_{31} \\ A\Lambda_{12}^2 + 2H\Lambda_{12}\Lambda_{31} + B\Lambda_{12}^2 = 1 \end{cases}$$

$$\Rightarrow \quad \Lambda_{12}^2 \left(\frac{2}{3m^2\sigma_\lambda^2} - 2 \cdot \frac{1}{3m^2\sigma_\lambda^2} + \frac{2}{3m^2\sigma_\lambda^2} \right) = 1$$

$$\Rightarrow \quad \begin{cases} \Lambda_{12} = \pm\sqrt{\dfrac{3}{2}} m\sigma_\lambda \\ \Lambda_{31} = \mp\sqrt{\dfrac{3}{2}} m\sigma_\lambda \end{cases}$$

Do último resultado conclui-se que os outros dois vértices da elipse Φ, que passam a chamar-se $\mathbf{V_{M1}}$ e $\mathbf{V_{M2}}$, possuem as seguintes coordenadas no referencial $\Lambda_{12}0'\Lambda_{31}$:

$$\begin{aligned} &\mathbf{V_{M1}} \left(\sqrt{\frac{3}{2}}m\sigma_\lambda, \quad -\sqrt{\frac{3}{2}}m\sigma_\lambda \right) \\ &\mathbf{V_{M2}} \left(-\sqrt{\frac{3}{2}}m\sigma_\lambda, \quad \sqrt{\frac{3}{2}}m\sigma_\lambda \right) \end{aligned} \tag{H.17}$$

De acordo com (H.9), as coordenadas destes vértices no referencial $\lambda_{12}0\lambda_{31}$ são as seguintes (Figura H.1):

$$\begin{aligned} &\mathbf{V_{M1}} \left(\lambda_{12m} + \sqrt{\frac{3}{2}}m\sigma_\lambda, \quad \lambda_{31m} - \sqrt{\frac{3}{2}}m\sigma_\lambda \right) \\ &\mathbf{V_{M2}} \left(\lambda_{12m} - \sqrt{\frac{3}{2}}m\sigma_\lambda, \quad \lambda_{31m} + \sqrt{\frac{3}{2}}m\sigma_\lambda \right) \end{aligned} \tag{H.18}$$

Os comprimentos dos semieixos da elipse Φ são dados por

$$\begin{cases} s_m = \sqrt{\dfrac{m^2\sigma_\lambda^2}{2} + \dfrac{m^2\sigma_\lambda^2}{2}} = m\sigma_\lambda \\ s_M = \sqrt{\dfrac{3}{2}m^2\sigma_\lambda^2 + \dfrac{3}{2}m^2\sigma_\lambda^2} = \sqrt{3}m\sigma_\lambda \end{cases} \tag{H.19}$$

- s_m é o comprimento de um dos semieixos que possui uma extremidade no vértice $\mathbf{V_{m1}}$ ou no vértice $\mathbf{V_{m2}}$;

- s_M é o comprimento de um dos semieixos que possui uma extremidade no vértice $\mathbf{V_{M1}}$ ou no vértice $\mathbf{V_{M2}}$.

Verifica-se que[5]

$$\begin{cases} \overline{\mathbf{V_{m1}\ F1}} = \overline{\mathbf{V_{m1}\ F2}} \\ \overline{\mathbf{V_{m1}\ F1}} + \overline{\mathbf{V_{m1}\ F2}} = 2s_M \\ \left(\overline{\mathbf{V_{m1}\ F1}}\right)^2 = \left(\overline{\mathbf{P0\ F1}}\right)^2 + s_m^2 \end{cases} \Rightarrow \begin{cases} \overline{\mathbf{V_{m1}\ F1}} = s_M = \sqrt{3}m\sigma_\lambda \\ \overline{\mathbf{P0\ F1}} = \sqrt{2}s_m = \sqrt{2}m\sigma_\lambda \end{cases} \tag{H.20}$$

[5] $\overline{\mathbf{V_{m1}\ F1}}$ é a distância entre os pontos $\mathbf{V_{m1}}$ e **F1**.

Assim, os focos da elipse Φ possuem as seguintes coordenadas no referencial $\lambda_{12}0\lambda_{31}$ (Figura H.1):

$$\begin{aligned} &\mathbf{F1}\ \left(\lambda_{12m} - s_m,\ \ \lambda_{31m} + s_m\right) \\ &\mathbf{F2}\ \left(\lambda_{12m} + s_m,\ \ \lambda_{31m} - s_m\right) \end{aligned} \tag{H.21}$$

ou seja,

$$\begin{aligned} &\mathbf{F1}\ \left(\lambda_{12m} - m\sigma_\lambda,\ \ \lambda_{31m} + m\sigma_\lambda\right) \\ &\mathbf{F2}\ \left(\lambda_{12m} + m\sigma_\lambda,\ \ \lambda_{31m} - m\sigma_\lambda\right) \end{aligned} \tag{H.22}$$

A elipse Φ encontra-se inscrita numa superfície de incerteza de medição hexagonal com quatro lados paralelos aos eixos coordenados do referencial $\lambda_{12}0\lambda_{31}$ e dois lados paralelos ao eixo maior da elipse (Figura H.1). Os vértices da superfície são os pontos **P1**, **P2**, **P3**, **P4**, **P5** e **P6**. A superfície é tangente à elipse nos pontos **P12**, **P23**, **P34**, **P45**, **P56** e **P61**. O passo seguinte consiste em determinar as coordenadas de todos estes pontos no referencial $\lambda_{12}0\lambda_{31}$.

Uma vez que **P2** se situa sobre a recta que contém os focos da elipse Φ, para determinar as suas coordenadas basta calcular a sua distância ao ponto **P0**.

A recta que passa pelos pontos **P2** e **P3** é tangente à elipse Φ no ponto **P23** e tem a seguinte equação no referencial $\alpha 0\beta$ (Figura H.1):

$$\alpha = \beta + \delta \tag{H.23}$$

em que δ é a abcissa de **P2** no referencial $\alpha 0\beta$.

Para calcular δ – cujo valor absoluto é a distância do ponto **P2** ao ponto **P0** – recorrendo a (H.23) é necessário obter as coordenadas de um dos pontos da recta que passa pelos pontos **P2** e **P3**, por exemplo o ponto **P23**.

A elipse Φ tem a seguinte equação no referencial $\alpha 0\beta$:

$$\frac{\alpha^2}{\left(\sqrt{3}s_m\right)^2} + \frac{\beta^2}{s_m^2} = 1 \tag{H.24}$$

Considerando apenas a parte da elipse tal que $\beta>0$,

$$\beta = \sqrt{s_m^2 - \frac{\alpha^2}{3}} \tag{H.25}$$

No ponto **P23** verifica-se que

$$\frac{d\beta}{d\alpha} = -\frac{1}{3}\frac{\alpha}{\sqrt{s_m^2 - \frac{\alpha^2}{3}}} = 1 \quad \Rightarrow \quad \begin{cases} \alpha = -\frac{3}{2}s_m \\ \beta = \frac{s_m}{2} \end{cases} \tag{H.26}$$

Substituindo estes valores de α e β em (H.23) obtém-se

$$-\frac{3}{2}s_m = \frac{s_m}{2} + \delta \quad \Rightarrow \qquad \delta = -2s_m \tag{H.27}$$

donde se conclui que a distância do ponto **P2** ao ponto **P0** é igual a $2s_m$. Assim, as coordenadas de **P2** no referencial $\lambda_{12}0\lambda_{31}$ são as seguintes:

$$\mathbf{P2}\ \left(\lambda_{12m} - \sqrt{2}s_m,\ \ \lambda_{31m} + \sqrt{2}s_m\right) \tag{H.28}$$

ou seja,

$$\mathbf{P2}\ \left(\lambda_{12m} - \sqrt{2}m\sigma_\lambda,\ \ \lambda_{31m} + \sqrt{2}m\sigma_\lambda\right) \tag{H.29}$$

A partir das coordenadas de **P2** e de **P23** no referencial $\alpha 0\beta$ é possível calcular a distância entre estes dois pontos, que é dada por

$$\overline{\mathbf{P2\ P23}} = \sqrt{\left[-\frac{3}{2}s_m - \left(-2s_m\right)\right]^2 + \left[\frac{s_m}{2} - 0\right]^2} = \frac{\sqrt{2}}{2}s_m = \frac{\sqrt{2}}{2}m\sigma_\lambda \tag{H.30}$$

As coordenadas de **P23** no referencial $\lambda_{12}0\lambda_{31}$ são as seguintes:

$$\mathbf{P23}\ \left(\lambda_{12m} - \frac{\sqrt{2}}{2}s_m,\ \ \lambda_{31m} + \sqrt{2}s_m\right) \tag{H.31}$$

ou seja,

$$\mathbf{P23}\left(\lambda_{12m}-\frac{\sqrt{2}}{2}m\sigma_{\lambda},\quad \lambda_{31m}+\sqrt{2}m\sigma_{\lambda}\right) \tag{H.32}$$

No ponto **P3**, intersecção das rectas $\alpha = \beta - 2s_m$ e $\beta = s_m$, verifica-se que

$$\begin{cases}\alpha=\beta-2s_m\\ \beta=s_m\end{cases}\quad\Rightarrow\quad\begin{cases}\alpha=-s_m\\ \beta=s_m\end{cases} \tag{H.33}$$

A distância entre **P2** e **P3** é dada por

$$\overline{\mathbf{P2\ P3}}=\sqrt{\left[-s_m-(-2s_m)\right]^2+\left[s_m-0\right]^2}=\sqrt{2}s_m=\sqrt{2}m\sigma_{\lambda} \tag{H.34}$$

donde se conclui que as coordenadas de **P3** no referencial $\lambda_{12}0\lambda_{31}$ são as seguintes:

$$\mathbf{P3}\left(\lambda_{12m},\quad \lambda_{31m}+\sqrt{2}s_m\right) \tag{H.35}$$

ou seja,

$$\mathbf{P3}\left(\lambda_{12m},\quad \lambda_{31m}+\sqrt{2}m\sigma_{\lambda}\right) \tag{H.36}$$

O ponto **P34** coincide com $\mathbf{V_{m1}}$, vértice da elipse Φ. Por isso, tem as seguintes coordenadas no referencial $\lambda_{12}0\lambda_{31}$:

$$\mathbf{P34}\left(\lambda_{12m}+\frac{\sqrt{2}}{2}m\sigma_{\lambda},\quad \lambda_{31m}+\frac{\sqrt{2}}{2}m\sigma_{\lambda}\right) \tag{H.37}$$

Uma vez determinadas as coordenadas de P2, P23, P3 e P34, é possível obter por simetria as coordenadas de todos os outros vértices da superfície de incerteza de medição na qual está inscrita a elipse Φ e dos pontos em que a elipse é tangente a essa superfície. As coordenadas, no referencial $\lambda_{12}0\lambda_{31}$, de todos os vértices e pontos de tangência são as seguintes:

$$\begin{aligned}
&\mathbf{P1}\ \left(\lambda_{12m}-\sqrt{2}s_m\ ,\ \lambda_{31m}\right)\\
&\mathbf{P2}\ \left(\lambda_{12m}-\sqrt{2}s_m\ ,\ \lambda_{31m}+\sqrt{2}s_m\right)\\
&\mathbf{P3}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}+\sqrt{2}s_m\right)\\
&\mathbf{P4}\ \left(\lambda_{12m}+\sqrt{2}s_m\ ,\ \lambda_{31m}\right)\\
&\mathbf{P5}\ \left(\lambda_{12m}+\sqrt{2}s_m\ ,\ \lambda_{31m}-\sqrt{2}s_m\right)\\
&\mathbf{P6}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}-\sqrt{2}s_m\right)
\end{aligned}
\quad\Leftrightarrow\quad
\begin{aligned}
&\mathbf{P1}\ \left(\lambda_{12m}-\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}\right)\\
&\mathbf{P2}\ \left(\lambda_{12m}-\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}+\sqrt{2}m\sigma_\lambda\right)\\
&\mathbf{P3}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}+\sqrt{2}m\sigma_\lambda\right)\\
&\mathbf{P4}\ \left(\lambda_{12m}+\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}\right)\\
&\mathbf{P5}\ \left(\lambda_{12m}+\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}-\sqrt{2}m\sigma_\lambda\right)\\
&\mathbf{P6}\ \left(\lambda_{12m}\ ,\ \lambda_{31m}-\sqrt{2}m\sigma_\lambda\right)
\end{aligned}
\qquad (H.38)$$

$$\begin{aligned}
&\mathbf{P12}\ \left(\lambda_{12m}-\sqrt{2}s_m\ ,\ \lambda_{31m}+\frac{\sqrt{2}}{2}s_m\right)\\
&\mathbf{P23}\ \left(\lambda_{12m}-\frac{\sqrt{2}}{2}s_m\ ,\ \lambda_{31m}+\sqrt{2}s_m\right)\\
&\mathbf{P34}\ \left(\lambda_{12m}+\frac{\sqrt{2}}{2}s_m\ ,\ \lambda_{31m}+\frac{\sqrt{2}}{2}s_m\right)\\
&\mathbf{P45}\ \left(\lambda_{12m}+\sqrt{2}s_m\ ,\ \lambda_{31m}-\frac{\sqrt{2}}{2}s_m\right)\\
&\mathbf{P56}\ \left(\lambda_{12m}+\frac{\sqrt{2}}{2}s_m\ ,\ \lambda_{31m}-\sqrt{2}s_m\right)\\
&\mathbf{P61}\ \left(\lambda_{12m}-\frac{\sqrt{2}}{2}s_m\ ,\ \lambda_{31m}-\frac{\sqrt{2}}{2}s_m\right)
\end{aligned}
\quad\Leftrightarrow\quad
\begin{aligned}
&\mathbf{P12}\ \left(\lambda_{12m}-\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}+\frac{\sqrt{2}}{2}m\sigma_\lambda\right)\\
&\mathbf{P23}\ \left(\lambda_{12m}-\frac{\sqrt{2}}{2}m\sigma_\lambda\ ,\ \lambda_{31m}+\sqrt{2}m\sigma_\lambda\right)\\
&\mathbf{P34}\ \left(\lambda_{12m}+\frac{\sqrt{2}}{2}m\sigma_\lambda\ ,\ \lambda_{31m}+\frac{\sqrt{2}}{2}m\sigma_\lambda\right)\\
&\mathbf{P45}\ \left(\lambda_{12m}+\sqrt{2}m\sigma_\lambda\ ,\ \lambda_{31m}-\frac{\sqrt{2}}{2}m\sigma_\lambda\right)\\
&\mathbf{P56}\ \left(\lambda_{12m}+\frac{\sqrt{2}}{2}m\sigma_\lambda\ ,\ \lambda_{31m}-\sqrt{2}m\sigma_\lambda\right)\\
&\mathbf{P61}\ \left(\lambda_{12m}-\frac{\sqrt{2}}{2}m\sigma_\lambda\ ,\ \lambda_{31m}-\frac{\sqrt{2}}{2}m\sigma_\lambda\right)
\end{aligned}
\qquad (H.39)$$

I. Código Fonte dos Programas de Teste

Neste anexo apresenta-se o código fonte, escrito em *Java 2*, dos programas utilizados nos quatro conjuntos de testes do Capítulo 6. Os parâmetros de cada programa correspondem ao primeiro teste de cada conjunto.

I.1 Código Fonte do Programa Usado no Primeiro Conjunto de Testes

```
import java.io.*;
import javax.vecmath.*;
class Ggt1
{
public static void main (String[] arguments)throws IOException
        {
        double x1, y1, x2, y2, x3, y3, xMIN, yMIN, xMAX, yMAX, dx, dy, casDec;
        double x, y, teta, l12, l23, l31;
        double xR, yR, tetaR;
        double lambda1, lambda2, lambda3, lamb12, lamb31;
        double fi, sigma, gama, delta, num, den, tau, l1, erroPos, erroOr;


        // Parâmetros ------------------------------------------------------------------------

        //COORDENADAS DAS BALIZAS
        x1 = 75;
        y1 = 75;
        x2 = 25;
        y2 = 60;
        x3 = 55;
        y3 = 25;

        //RECINTO DE NAVEGAÇÃO
        xMIN = 0;
        yMIN = 0;
        xMAX = 100;
        yMAX = 100;

        //INCREMENTOS
        dx = 0.1;
        dy = 0.1;

        casDec = 0;


        // CÁLCULO DE l12 E l31 --------------------------------------------------------------

        Vector3d v12 = new Vector3d(x2-x1, y2-y1, 0);
        Vector3d v31 = new Vector3d(x1-x3, y1-y3, 0);

        l12 = v12.length();
        l31 = v31.length();


        // CÁLCULO DE fi ---------------------------------------------------------------------

        Vector3d v1 = new Vector3d(-1,0,0);

        fi = v1.angle(v12);

        Vector3d v112 = new Vector3d();
        v112.cross(v1,v12);
        if(v112.z < 0)
                fi = -fi;
```

```
// CÁLCULO DE sigma ----------------------------------------------------------------------

sigma = v31.angle(v12);

Vector3d v3112 = new Vector3d();
v3112.cross(v31,v12);
if(v3112.z < 0)
        sigma = -sigma;


// CÁLCULO DE delta ----------------------------------------------------------------------

Vector3d v21 = new Vector3d(x1-x2, y1-y2, 0);
Vector3d v23 = new Vector3d(x3-x2, y3-y2, 0);
delta = v21.angle(v23);
if(sigma < 0)
        delta = -delta;


// SAÍDAS --------------------------------------------------------------------------------

PrintStream ps1 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/X.dat"));
PrintStream ps2 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/Y.dat"));
PrintStream ps3 = new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/erroPos/Z.dat"));
PrintStream ps4 = new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/erroOr/Z.dat"));


PrintStream ps5 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/GGT1.txt"));
ps5.println("x1 = " + x1);
ps5.println("y1 = " + y1);
ps5.println("x2 = " + x2);
ps5.println("y2 = " + y2);
ps5.println("x3 = " + x3);
ps5.println("y3 = " + y3);
ps5.println("xMIN = " + xMIN);
ps5.println("yMIN = " + yMIN);
ps5.println("xMAX = " + xMAX);
ps5.println("yMAX = " + yMAX);
ps5.println("dx = " + dx);
ps5.println("dy = " + dy);
ps5.println("l12 = " + l12);
ps5.println("l31 = " + l31);
ps5.println("fi = " + Math.toDegrees(fi) + "º");
ps5.println("sigma = " + Math.toDegrees(sigma) + "º");
ps5.println("delta = " + Math.toDegrees(delta) + "º");

//-----------------------------------------------------------------------------------------


Vector3d vHoriz = new Vector3d(1,0,0);

y = yMIN;
while(y<=yMAX)
{
        x = xMIN;
        while(x<=xMAX)
        {
                //TETA

                Vector3d vR0 = new Vector3d(Math.pow(-1, Math.rint(Math.random()*10))
                        * Math.random()*10, Math.pow(-1, Math.rint(Math.random()*10))
                        * Math.random()*10, 0);

                if(vR0.length() == 0)
                        vR0.x = 1;

                teta = vHoriz.angle(vR0);
                Vector3d vAux = new Vector3d();
                vAux.cross(vHoriz,vR0);
                if(vAux.z < 0)
                        teta = -teta;


                // CÁLCULO DE lambda12 E lambda31 ---------------------------------------

                Vector3d vR1 = new Vector3d(x1-x, y1-y, 0);
                Vector3d vR2 = new Vector3d(x2-x, y2-y, 0);
                Vector3d vR3 = new Vector3d(x3-x, y3-y, 0);


                lambda1 = vR0.angle(vR1);
                Vector3d vR0R1 = new Vector3d();
                vR0R1.cross(vR0,vR1);
                if(vR0R1.z < 0)
                        lambda1 = Math.PI*2 - lambda1;
```

```
lambda2 = vR0.angle(vR2);
Vector3d vR0R2 = new Vector3d();
vR0R2.cross(vR0,vR2);
if(vR0R2.z < 0)
        lambda2 = Math.PI*2 - lambda2;

lambda3 = vR0.angle(vR3);
Vector3d vR0R3 = new Vector3d();
vR0R3.cross(vR0,vR3);
if(vR0R3.z < 0)
        lambda3 = Math.PI*2 - lambda3;


lambda1 = Math.toRadians
                  (Math.rint(Math.pow(10,casDec)*Math.toDegrees(lambda1))
                  /Math.pow(10,casDec));

lambda2 = Math.toRadians
                  (Math.rint(Math.pow(10,casDec)*Math.toDegrees(lambda2))
                  /Math.pow(10,casDec));

lambda3 = Math.toRadians
                  (Math.rint(Math.pow(10,casDec)*Math.toDegrees(lambda3))
                  /Math.pow(10,casDec));


lamb12 = lambda2 - lambda1;
if(lambda2 < lambda1)
        lamb12 += Math.PI*2;


lamb31 = lambda1 - lambda3;
if(lambda1 < lambda3)
        lamb31 += Math.PI*2;



// CÁLCULO DE xR, yR e tetaR ------------------------------------------------


        gama = sigma - lamb31;

        num = Math.sin(lamb12)*(l12*Math.sin(lamb31)-l31*Math.sin(gama));
        den = l31*Math.sin(lamb12)*
                Math.cos(gama)-l12*Math.cos(lamb12)*Math.sin(lamb31);
        tau = Math.atan(num/den);


        if((lamb12 < Math.PI) && (tau < 0))
                tau = tau + Math.PI;
        if((lamb12 > Math.PI) && (tau > 0))
                tau = tau - Math.PI;

        if(tau == 0 && ((sigma<0 && lamb31<Math.PI) ||
                        (sigma>0 && lamb31>Math.PI)))
                tau = Math.PI;


        if(Math.abs(Math.sin(lamb12)) > Math.abs(Math.sin(lamb31)))
                l1 = l12*Math.sin(tau + lamb12)/Math.sin(lamb12);
        else
                l1 = l31*Math.sin(tau + sigma - lamb31)/Math.sin(lamb31);



        xR = x1-l1*Math.cos(fi+tau);
        yR = y1-l1*Math.sin(fi+tau);

        tetaR = fi+tau-lambda1;

        if(tetaR <= -Math.PI)
                tetaR = tetaR + Math.PI*2;
        if(tetaR > Math.PI)
                tetaR = tetaR - Math.PI*2;



        // CÁLCULO DE ERROS ---------------------------------------------


        // ERRO DE POSIÇÃO

        Vector2d vPos = new Vector2d(x, y);
        Vector2d vErroPos = new Vector2d(xR-x, yR-y);
                erroPos = vErroPos.length();
```

```
                                // ERRO DE ORIENTAÇÃO

                                erroOr = Math.toDegrees(Math.abs(tetaR-teta));
                                if (erroOr > 180)
                                        erroOr = 360 - erroOr;


                        // SAÍDAS ----------------------------------------------------------------

                        // X e Y
                        ps1.print(x + "\t");
                        ps2.print(y + "\t");

                        // ERRO DE POSIÇÃO
                        ps3.print(erroPos + "\t");

                        // ERRO DE ORIENTAÇÃO
                        ps4.print(erroOr + "\t");

                        //-------------------------------------------------------------------------

                        x += dx;
                }
                ps1.println("");
                ps2.println("");
                ps3.println("");
                ps4.println("");

                y += dy;
        }
        ps1.close();
        ps2.close();
        ps3.close();
        ps4.close();
        ps5.close();
        }
}
```

I.2 Código Fonte do Programa Usado no Segundo Conjunto de Testes

```
import java.io.*;
import javax.vecmath.*;
class Ggt2
{
public static void main (String[] arguments)throws IOException
        {
        double x1, y1, x2, y2, x3, y3, xMIN, yMIN, xMAX, yMAX, dx, dy, casDec;
        double x, y, teta, l12, l23, l31;
        double xR, yR, tetaR;
        double lambda1, lambda2, lambda3, lamb12, lamb31;
        double fi, sigma, gama, delta, num, den, tau, l1, erroPos, erroOr, erroPosMax, erroOrMax;


        // Parâmetros ------------------------------------------------------------------------

        //COORDENADAS DAS BALIZAS
        x1 = 75;
        y1 = 75;
        x2 = 25;
        y2 = 60;
        x3 = 55;
        y3 = 25;


        //RECINTO DE NAVEGAÇÃO
        xMIN = 0;
        yMIN = 0;
        xMAX = 100;
        yMAX = 100;

        //INCREMENTOS
        dx = 0.1;
        dy = 0.1;

        casDec=2;
        erroPosMax = 0.05;
        erroOrMax = 0.06;
```

```
// CÁLCULO DE l12 E l31 ------------------------------------------------------------------------

Vector3d v12 = new Vector3d(x2-x1, y2-y1, 0);
Vector3d v31 = new Vector3d(x1-x3, y1-y3, 0);

l12 = v12.length();
l31 = v31.length();


// CÁLCULO DE fi -------------------------------------------------------------------------------

Vector3d v1 = new Vector3d(-1,0,0);

fi = v1.angle(v12);

Vector3d v112 = new Vector3d();
v112.cross(v1,v12);
if(v112.z < 0)
        fi = -fi;


// CÁLCULO DE sigma ----------------------------------------------------------------------------

sigma = v31.angle(v12);

Vector3d v3112 = new Vector3d();
v3112.cross(v31,v12);
if(v3112.z < 0)
        sigma = -sigma;


// CÁLCULO DE delta ----------------------------------------------------------------------------

Vector3d v21 = new Vector3d(x1-x2, y1-y2, 0);
Vector3d v23 = new Vector3d(x3-x2, y3-y2, 0);
delta = v21.angle(v23);
if(sigma < 0)
        delta = -delta;


// SAÍDAS --------------------------------------------------------------------------------------

PrintStream ps1 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/X.dat"));
PrintStream ps2 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/Y.dat"));
PrintStream ps3 = new PrintStream
                (new FileOutputStream("c:/jse/trabalho/result/erroPos/Z.dat"));
PrintStream ps4 = new PrintStream
                (new FileOutputStream("c:/jse/trabalho/result/erroOr/Z.dat"));


PrintStream ps5 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/GGT2.txt"));
ps5.println("x1 = " + x1);
ps5.println("y1 = " + y1);
ps5.println("x2 = " + x2);
ps5.println("y2 = " + y2);
ps5.println("x3 = " + x3);
ps5.println("y3 = " + y3);
ps5.println("xMIN = " + xMIN);
ps5.println("yMIN = " + yMIN);
ps5.println("xMAX = " + xMAX);
ps5.println("yMAX = " + yMAX);
ps5.println("dx = " + dx);
ps5.println("dy = " + dy);
ps5.println("l12 = " + l12);
ps5.println("l31 = " + l31);
ps5.println("fi = " + Math.toDegrees(fi) + "º");
ps5.println("sigma = " + Math.toDegrees(sigma) + "º");
ps5.println("delta = " + Math.toDegrees(delta) + "º");

//-----------------------------------------------------------------------------------------------



Vector3d vHoriz = new Vector3d(1,0,0);

y = yMIN;
while(y<=yMAX)
{
        x = xMIN;
        while(x<=xMAX)
        {
                //TETA

                Vector3d vR0 = new Vector3d(Math.pow(-1, Math.rint(Math.random()*10))
                        * Math.random()*10, Math.pow(-1, Math.rint(Math.random()*10))
                        * Math.random()*10, 0);
```

```
if(vR0.length() == 0)
        vR0.x = 1;

teta = vHoriz.angle(vR0);
Vector3d vAux = new Vector3d();
vAux.cross(vHoriz,vR0);
if(vAux.z < 0)
        teta = -teta;


// CÁLCULO DE lambda12 E lambda31 ----------------------------------------

Vector3d vR1 = new Vector3d(x1-x, y1-y, 0);
Vector3d vR2 = new Vector3d(x2-x, y2-y, 0);
Vector3d vR3 = new Vector3d(x3-x, y3-y, 0);


lambda1 = vR0.angle(vR1);
Vector3d vR0R1 = new Vector3d();
vR0R1.cross(vR0,vR1);
if(vR0R1.z < 0)
        lambda1 = Math.PI*2 - lambda1;

lambda2 = vR0.angle(vR2);
Vector3d vR0R2 = new Vector3d();
vR0R2.cross(vR0,vR2);
if(vR0R2.z < 0)
        lambda2 = Math.PI*2 - lambda2;

lambda3 = vR0.angle(vR3);
Vector3d vR0R3 = new Vector3d();
vR0R3.cross(vR0,vR3);
if(vR0R3.z < 0)
        lambda3 = Math.PI*2 - lambda3;


lambda1 = Math.toRadians
                (Math.rint(Math.pow(10,casDec)*Math.toDegrees(lambda1))
                /Math.pow(10,casDec));

lambda2 = Math.toRadians
                (Math.rint(Math.pow(10,casDec)*Math.toDegrees(lambda2))
                /Math.pow(10,casDec));

lambda3 = Math.toRadians
                (Math.rint(Math.pow(10,casDec)*Math.toDegrees(lambda3))
                /Math.pow(10,casDec));


lamb12 = lambda2 - lambda1;
if(lambda2 < lambda1)
        lamb12 += Math.PI*2;


lamb31 = lambda1 - lambda3;
if(lambda1 < lambda3)
        lamb31 += Math.PI*2;



// CÁLCULO DE xR, yR e tetaR ---------------------------------------------


        gama = sigma - lamb31;

        num = Math.sin(lamb12)*(l12*Math.sin(lamb31)-l31*Math.sin(gama));
        den = l31*Math.sin(lamb12)
                 *Math.cos(gama)-l12*Math.cos(lamb12)*Math.sin(lamb31);
        tau = Math.atan(num/den);


        if((lamb12 < Math.PI) && (tau < 0))
                tau = tau + Math.PI;
        if((lamb12 > Math.PI) && (tau > 0))
                tau = tau - Math.PI;

        if(tau == 0 && ((sigma<0 && lamb31<Math.PI) ||
                         (sigma>0 && lamb31>Math.PI)))
                tau = Math.PI;


        if(Math.abs(Math.sin(lamb12)) > Math.abs(Math.sin(lamb31)))
                l1 = l12*Math.sin(tau + lamb12)/Math.sin(lamb12);
        else
                l1 = l31*Math.sin(tau + sigma - lamb31)/Math.sin(lamb31);
```

```
                                xR = x1-l1*Math.cos(fi+tau);
                                yR = y1-l1*Math.sin(fi+tau);

                                tetaR = fi+tau-lambda1;

                                if(tetaR <= -Math.PI)
                                        tetaR = tetaR + Math.PI*2;
                                if(tetaR > Math.PI)
                                        tetaR = tetaR - Math.PI*2;



                                // CÁLCULO DE ERROS ------------------------------------------


                                // ERRO DE POSIÇÃO

                                Vector2d vPos = new Vector2d(x, y);
                                Vector2d vErroPos = new Vector2d(xR-x, yR-y);
                                        erroPos = vErroPos.length();


                                // ERRO DE ORIENTAÇÃO

                                erroOr = Math.toDegrees(Math.abs(tetaR-teta));
                                if (erroOr > 180)
                                        erroOr = 360 - erroOr;



                        // SAÍDAS ----------------------------------------------------------------

                        // X e Y
                        ps1.print(x + "\t");
                        ps2.print(y + "\t");

                        // ERRO DE POSIÇÃO
                        if(erroPos > erroPosMax)
                                erroPos = erroPosMax;
                        ps3.print(erroPos + "\t");

                        // ERRO DE ORIENTAÇÃO
                        if(erroOr > erroOrMax)
                                erroOr = erroOrMax;
                        ps4.print(erroOr + "\t");

                        //-------------------------------------------------------------------------

                        x += dx;
                }
                ps1.println("");
                ps2.println("");
                ps3.println("");
                ps4.println("");

                y += dy;
        }
        ps1.close();
        ps2.close();
        ps3.close();
        ps4.close();
        ps5.close();
        }
}
```

I.3 Código Fonte do Programa Usado no Terceiro Conjunto de Testes

```
import java.io.*;
import javax.vecmath.*;
class Ggt3
{
public static void main (String[] arguments)throws IOException
        {
        double x1, y1, x2, y2, x3, y3, xMIN, yMIN, xMAX, yMAX, dx, dy, casDec;
        double x, y, teta, l12, l23, l31, lambda1, lambda2, lambda3, lamb12, lamb23, lamb31,
                deltaLambda;
        double xR[]=new double [7];
        double yR[]=new double [7];
```

```
double tetaR;
double fi, sigma, gama, delta, num, den, tau, l1;
double erroPos, erroOr, erroPosEstimado, deltaErroPos, erroOrEstimado, deltaErroOr;
double erroPosMax, erroOrMax, erroPosEstimadoMax, deltaErroPosMax, erroOrEstimadoMax,
        deltaErroOrMax;
double dist30, dist40, dist50, dist60, dist10, dist20, dist70, dist80;
double dist12, dist23, dist34, dist45, dist56, dist61, distMax, dmax;
double deltaTau10, deltaTau20, deltaTau30, deltaTau40, deltaTau50, deltaTau60;
double deltaTauTA, deltaTauTB, deltaTauMax;
double ang12, ang23, ang34, ang45, ang56, ang61;
double f;
double w, rT, xCL,yCL, distCLB1, distCLB2, distCLB3;
double lambdaPi, psi, R, lCM23, lCM, lC1, lTM, uC1x, uC1y, uMTAx, uMTAy;
double xC, yC, xM23, yM23, xM, yM, xTA, yTA, xTB, yTB, lambda12TA, lambda12TB;



// Parâmetros ------------------------------------------------------------------------


//COORDENADAS DAS BALIZAS
x1 = 75;
y1 = 75;
x2 = 25;
y2 = 60;
x3 = 55;
y3 = 25;

//RECINTO DE NAVEGAÇÃO
xMIN = 0;
yMIN = 0;
xMAX = 100;
yMAX = 100;

//INCREMENTOS
dx = 0.1;
dy = 0.1;

casDec=0;
erroPosMax = 5;
erroOrMax = 6;
deltaLambda = Math.toRadians(0.5);

erroPosEstimadoMax = erroPosMax;
deltaErroPosMax = erroPosMax;
erroOrEstimadoMax = erroOrMax;
deltaErroOrMax = erroOrMax;


// CÁLCULO DE l12 E l31 --------------------------------------------------------------

Vector3d v12 = new Vector3d(x2-x1, y2-y1, 0);
Vector3d v31 = new Vector3d(x1-x3, y1-y3, 0);

l12 = v12.length();
l31 = v31.length();


// CÁLCULO DE fi ---------------------------------------------------------------------

Vector3d v1 = new Vector3d(-1,0,0);

fi = v1.angle(v12);

Vector3d v112 = new Vector3d();
v112.cross(v1,v12);
if(v112.z < 0)
        fi = -fi;


// CÁLCULO DE sigma ------------------------------------------------------------------

sigma = v31.angle(v12);

Vector3d v3112 = new Vector3d();
v3112.cross(v31,v12);
if(v3112.z < 0)
        sigma = -sigma;


// CÁLCULO DE delta ------------------------------------------------------------------

Vector3d v21 = new Vector3d(x1-x2, y1-y2, 0);
Vector3d v23 = new Vector3d(x3-x2, y3-y2, 0);
delta = v21.angle(v23);
if(sigma < 0)
        delta = -delta;
```

```
// SAÍDAS-------------------------------------------------------------------------------

PrintStream ps1 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/X.dat"));
PrintStream ps2 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/Y.dat"));
PrintStream ps3= new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/erroPos/Z.dat"));
PrintStream ps4 = new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/erroOr/Z.dat"));
PrintStream ps5 = new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/erroPosEstimado/Z.dat"));
PrintStream ps6 = new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/deltaErroPos/Z.dat"));
PrintStream ps7 = new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/erroOrEstimado/Z.dat"));
PrintStream ps8 = new PrintStream
                 (new FileOutputStream("c:/jse/trabalho/result/deltaErroOr/Z.dat"));

PrintStream ps9 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/GGT3.txt"));
ps9.println("x1 = " + x1);
ps9.println("y1 = " + y1);
ps9.println("x2 = " + x2);
ps9.println("y2 = " + y2);
ps9.println("x3 = " + x3);
ps9.println("y3 = " + y3);
ps9.println("xMIN = " + xMIN);
ps9.println("yMIN = " + yMIN);
ps9.println("xMAX = " + xMAX);
ps9.println("yMAX = " + yMAX);
ps9.println("dx = " + dx);
ps9.println("dy = " + dy);
ps9.println("casDec = " + casDec);
ps9.println("l12 = " + l12);
ps9.println("l31 = " + l31);
ps9.println("fi = " + Math.toDegrees(fi) + "º");
ps9.println("sigma = " + Math.toDegrees(sigma) + "º");
ps9.println("delta = " + Math.toDegrees(delta) + "º");
ps9.println("erroPosMax = " + erroPosMax);
ps9.println("erroOrMax = " + erroOrMax + "º");
ps9.println("deltaLambda = " + deltaLambda + "rad = " + Math.toDegrees(deltaLambda) + "º");
ps9.println("erroPosEstimadoMax = " + erroPosEstimadoMax);
ps9.println("deltaErroPosMax = " + deltaErroPosMax);

//-------------------------------------------------------------------------------------



Vector3d vHoriz = new Vector3d(1,0,0);

y = yMIN;
while(y<=yMAX)
{
        x = xMIN;
        while(x<=xMAX)
        {
                //TETA
                Vector3d vR0 = new Vector3d(Math.pow(-1, Math.rint(Math.random()*10))
                        * Math.random()*10, Math.pow(-1, Math.rint(Math.random()*10))
                        * Math.random()*10, 0);

                if(vR0.length() == 0)
                        vR0.x = 1;

                teta = vHoriz.angle(vR0);
                Vector3d vAux = new Vector3d();
                vAux.cross(vHoriz,vR0);
                if(vAux.z < 0)
                        teta = -teta;


                // CÁLCULO DE lambda12 E lambda31-----------------------------------------

                Vector3d vR1 = new Vector3d(x1-x, y1-y, 0);
                Vector3d vR2 = new Vector3d(x2-x, y2-y, 0);
                Vector3d vR3 = new Vector3d(x3-x, y3-y, 0);


                lambda1 = vR0.angle(vR1);
                Vector3d vR0R1 = new Vector3d();
                vR0R1.cross(vR0,vR1);
                if(vR0R1.z < 0)
                        lambda1 = Math.PI*2 - lambda1;

                lambda2 = vR0.angle(vR2);
                Vector3d vR0R2 = new Vector3d();
                vR0R2.cross(vR0,vR2);
                if(vR0R2.z < 0)
                        lambda2 = Math.PI*2 - lambda2;
```

```
lambda3 = vR0.angle(vR3);
Vector3d vR0R3 = new Vector3d();
vR0R3.cross(vR0,vR3);
if(vR0R3.z < 0)
        lambda3 = Math.PI*2 - lambda3;


//INTRODUZIR ERROS

lambda1 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                *Math.toDegrees(lambda1))/Math.pow(10,casDec));

lambda2 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                *Math.toDegrees(lambda2))/Math.pow(10,casDec));

lambda3 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                *Math.toDegrees(lambda3))/Math.pow(10,casDec));


lamb12 = lambda2 - lambda1;
if(lambda2 < lambda1)
        lamb12 += Math.PI*2;

lamb31 = lambda1 - lambda3;
if(lambda1 < lambda3)
        lamb31 += Math.PI*2;

lamb23 = lambda3 - lambda2;
if(lambda3 < lambda2)
        lamb23 += Math.PI*2;



if(Math.abs(lamb12+lamb31-sigma-Math.PI)<=2*deltaLambda
        || Math.abs(lamb12+lamb31-sigma-3*Math.PI)<=2*deltaLambda
        || (sigma<0 &&
        (Math.abs(lamb12-sigma+delta-Math.PI*2)<=2*deltaLambda
                || Math.abs(lamb31-delta-Math.PI*2)<=2*deltaLambda))
        || (sigma>=0 &&
        (Math.abs(lamb12-sigma+delta)<=2*deltaLambda
                || Math.abs(lamb31-delta)<=2*deltaLambda)))
{
        erroPos = Math.acos(2);
        erroOr = Math.acos(2);
        erroPosEstimado = Math.acos(2);
        deltaErroPos = Math.acos(2);
        erroOrEstimado = Math.acos(2);
        deltaErroOr = Math.acos(2);
}

else
{

// CÁLCULO DA POSIÇÃO E DA ORIENTAÇÃO DO ROBÔ ----------------------------

double lambda12[]={lamb12, lamb12-2*deltaLambda, lamb12-2*deltaLambda,
        lamb12, lamb12+2*deltaLambda, lamb12+2*deltaLambda, lamb12};
double lambda31[]={lamb31, lamb31, lamb31+2*deltaLambda,
        lamb31+2*deltaLambda, lamb31, lamb31-2*deltaLambda,
        lamb31-2*deltaLambda};


        gama = sigma - lambda31[0];

        num = Math.sin(lambda12[0])*(l12*Math.sin(lambda31[0])
                -l31*Math.sin(gama));
        den = l31*Math.sin(lambda12[0])*Math.cos(gama)
                -l12*Math.cos(lambda12[0])*Math.sin(lambda31[0]);
        tau = Math.atan(num/den);

        if((lambda12[0] < Math.PI) && (tau < 0))
                tau = tau + Math.PI;
        if((lambda12[0] > Math.PI) && (tau > 0))
                tau = tau - Math.PI;

        if(tau == 0 && ((sigma<0 && lambda31[0]<Math.PI)
                        || (sigma>0 && lambda31[0]>Math.PI)))
                tau = Math.PI;

        if(Math.abs(Math.sin(lambda12[0])) >
                                Math.abs(Math.sin(lambda31[0])))
                l1 = l12*Math.sin(tau + lambda12[0])
                                        /Math.sin(lambda12[0]);
        else
                l1 = l31*Math.sin(tau + sigma - lambda31[0])
                                        /Math.sin(lambda31[0]);
```

```
        xR[0] = x1-l1*Math.cos(fi+tau);
        yR[0] = y1-l1*Math.sin(fi+tau);

        tetaR = fi+tau-lambda1;

        if(tetaR <= -Math.PI)
                tetaR = tetaR + Math.PI*2;
        if(tetaR > Math.PI)
                tetaR = tetaR - Math.PI*2;


// CÁLCULO DAS COORDENADAS DOS VÉRTICES DA SUP. DE INCERTEZA DE POSIÇÃO ---

for (int i=1; i<7; i++)
        {
        gama = sigma - lambda31[i];

        num = Math.sin(lambda12[i])*(l12*Math.sin(lambda31[i])
                -l31*Math.sin(gama));
        den = l31*Math.sin(lambda12[i])*Math.cos(gama)
                -l12*Math.cos(lambda12[i])*Math.sin(lambda31[i]);
        tau = Math.atan(num/den);

        if((lambda12[i] < Math.PI) && (tau < 0))
                tau = tau + Math.PI;
        if((lambda12[i] > Math.PI) && (tau > 0))
                tau = tau - Math.PI;

        if(tau == 0 && ((sigma<0 && lambda31[i]<Math.PI)
                        || (sigma>0 && lambda31[i]>Math.PI)))
                tau = Math.PI;

        if(Math.abs(Math.sin(lambda12[i])) >
                                Math.abs(Math.sin(lambda31[i])))
                l1 = l12*Math.sin(tau + lambda12[i])
                                        /Math.sin(lambda12[i]);
        else
                l1 = l31*Math.sin(tau + sigma - lambda31[i])
                                        /Math.sin(lambda31[i]);

        xR[i] = x1-l1*Math.cos(fi+tau);
        yR[i] = y1-l1*Math.sin(fi+tau);
        }


        //CÁLCULO DE distMax ------------------------------------------

        Vector3d vP0P1 = new Vector3d(xR[1]-xR[0], yR[1]-yR[0], 0);
        Vector3d vP0P2 = new Vector3d(xR[2]-xR[0], yR[2]-yR[0], 0);
        Vector3d vP0P3 = new Vector3d(xR[3]-xR[0], yR[3]-yR[0], 0);
        Vector3d vP0P4 = new Vector3d(xR[4]-xR[0], yR[4]-yR[0], 0);
        Vector3d vP0P5 = new Vector3d(xR[5]-xR[0], yR[5]-yR[0], 0);
        Vector3d vP0P6 = new Vector3d(xR[6]-xR[0], yR[6]-yR[0], 0);

        dist10 = vP0P1.length();
        dist20 = vP0P2.length();
        dist30 = vP0P3.length();
        dist40 = vP0P4.length();
        dist50 = vP0P5.length();
        dist60 = vP0P6.length();

        Vector3d sub12 = new Vector3d();
        sub12.sub(vP0P2,vP0P1);
        dist12 = sub12.length();

        Vector3d sub23 = new Vector3d();
        sub23.sub(vP0P3,vP0P2);
        dist23 = sub23.length();

        Vector3d sub34 = new Vector3d();
        sub34.sub(vP0P4,vP0P3);
        dist34 = sub34.length();

        Vector3d sub45 = new Vector3d();
        sub45.sub(vP0P5,vP0P4);
        dist45 = sub45.length();

        Vector3d sub56 = new Vector3d();
        sub56.sub(vP0P6,vP0P5);
        dist56 = sub56.length();

        Vector3d sub61 = new Vector3d();
        sub61.sub(vP0P1,vP0P6);
        dist61 = sub61.length();

        ang12 = vP0P1.angle(vP0P2);
        ang23 = vP0P2.angle(vP0P3);
        ang34 = vP0P3.angle(vP0P4);
```

```
ang45 = vP0P4.angle(vP0P5);
ang56 = vP0P5.angle(vP0P6);
ang61 = vP0P6.angle(vP0P1);

distMax = 0;
if (dist10 > distMax)
        distMax = dist10;
if (dist20 > distMax)
        distMax = dist20;
if (dist30 > distMax)
        distMax = dist30;
if (dist40 > distMax)
        distMax = dist40;
if (dist50 > distMax)
        distMax = dist50;
if (dist60 > distMax)
        distMax = dist60;

if (dist10 > dist20)
        {
        w = Math.asin(Math.sin(ang12)*dist20/dist12);
        rT = (dist12/2)/Math.cos(w);
        xCL = xR[1] - (rT/dist10) * (xR[1] - xR[0]);
        yCL = yR[1] - (rT/dist10) * (yR[1] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang12)*dist10/dist12);
        rT = (dist12/2)/Math.cos(w);
        xCL = xR[2] - (rT/dist20) * (xR[2] - xR[0]);
        yCL = yR[2] - (rT/dist20) * (yR[2] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));

if (distCLB1 < rT || distCLB2 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)+
                                Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist20 > dist30)
        {
        w = Math.asin(Math.sin(ang23)*dist30/dist23);
        rT = (dist23/2)/Math.cos(w);
        xCL = xR[2] - (rT/dist20) * (xR[2] - xR[0]);
        yCL = yR[2] - (rT/dist20) * (yR[2] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang23)*dist20/dist23);
        rT = (dist23/2)/Math.cos(w);
        xCL = xR[3] - (rT/dist30) * (xR[3] - xR[0]);
        yCL = yR[3] - (rT/dist30) * (yR[3] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));

if (distCLB1 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)+
                                Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist30 > dist40)
        {
        w = Math.asin(Math.sin(ang34)*dist40/dist34);
        rT = (dist34/2)/Math.cos(w);
        xCL = xR[3] - (rT/dist30) * (xR[3] - xR[0]);
        yCL = yR[3] - (rT/dist30) * (yR[3] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang34)*dist30/dist34);
        rT = (dist34/2)/Math.cos(w);
        xCL = xR[4] - (rT/dist40) * (xR[4] - xR[0]);
        yCL = yR[4] - (rT/dist40) * (yR[4] - yR[0]);
        }

distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));
```

```
if (distCLB2 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)+
                                Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist40 > dist50)
        {
        w = Math.asin(Math.sin(ang45)*dist50/dist45);
        rT = (dist45/2)/Math.cos(w);
        xCL = xR[4] - (rT/dist40) * (xR[4] - xR[0]);
        yCL = yR[4] - (rT/dist40) * (yR[4] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang45)*dist40/dist45);
        rT = (dist45/2)/Math.cos(w);
        xCL = xR[5] - (rT/dist50) * (xR[5] - xR[0]);
        yCL = yR[5] - (rT/dist50) * (yR[5] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));

if (distCLB1 < rT || distCLB2 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)+
                                Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist50 > dist60)
        {
        w = Math.asin(Math.sin(ang56)*dist60/dist56);
        rT = (dist56/2)/Math.cos(w);
        xCL = xR[5] - (rT/dist50) * (xR[5] - xR[0]);
        yCL = yR[5] - (rT/dist50) * (yR[5] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang56)*dist50/dist56);
        rT = (dist56/2)/Math.cos(w);
        xCL = xR[6] - (rT/dist60) * (xR[6] - xR[0]);
        yCL = yR[6] - (rT/dist60) * (yR[6] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));

if (distCLB1 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)+
                                Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist60 > dist10)
        {
        w = Math.asin(Math.sin(ang61)*dist10/dist61);
        rT = (dist61/2)/Math.cos(w);
        xCL = xR[1] - (rT/dist60) * (xR[1] - xR[0]);
        yCL = yR[1] - (rT/dist60) * (yR[1] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang61)*dist60/dist61);
        rT = (dist61/2)/Math.cos(w);
        xCL = xR[1] - (rT/dist10) * (xR[1] - xR[0]);
        yCL = yR[1] - (rT/dist10) * (yR[1] - yR[0]);
        }

distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));

if (distCLB2 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)+
                                Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }
```

```
//CÁLCULO DE deltaTauMax ----------------------------------------

Vector3d vB1P0 = new Vector3d(xR[0]-x1, yR[0]-y1, 0);
Vector3d vB1P1 = new Vector3d(xR[1]-x1, yR[1]-y1, 0);
Vector3d vB1P2 = new Vector3d(xR[2]-x1, yR[2]-y1, 0);
Vector3d vB1P3 = new Vector3d(xR[3]-x1, yR[3]-y1, 0);
Vector3d vB1P4 = new Vector3d(xR[4]-x1, yR[4]-y1, 0);
Vector3d vB1P5 = new Vector3d(xR[5]-x1, yR[5]-y1, 0);
Vector3d vB1P6 = new Vector3d(xR[6]-x1, yR[6]-y1, 0);

deltaTau10 = vB1P1.angle(vB1P0);
deltaTau20 = vB1P2.angle(vB1P0);
deltaTau30 = vB1P3.angle(vB1P0);
deltaTau40 = vB1P4.angle(vB1P0);
deltaTau50 = vB1P5.angle(vB1P0);
deltaTau60 = vB1P6.angle(vB1P0);

deltaTauMax = 0;

if (deltaTau10 > deltaTauMax)
        deltaTauMax = deltaTau10;
if (deltaTau20 > deltaTauMax)
        deltaTauMax = deltaTau20;
if (deltaTau30 > deltaTauMax)
        deltaTauMax = deltaTau30;
if (deltaTau40 > deltaTauMax)
        deltaTauMax = deltaTau40;
if (deltaTau50 > deltaTauMax)
        deltaTauMax = deltaTau50;
if (deltaTau60 > deltaTauMax)
        deltaTauMax = deltaTau60;

//SE f=0 ENTÃO NÃO PODE HAVER TG

f = 0;
if((sigma>0 && sigma<Math.PI &&
   ((lamb12>Math.PI-2*deltaLambda &&
        lamb12<sigma-delta+Math.PI-2*deltaLambda)
   ||(lamb31>Math.PI-2*deltaLambda &&
        lamb31<delta+Math.PI-2*deltaLambda)
   ||(lamb12+lamb31>2*Math.PI+2*deltaLambda &&
        lamb12+lamb31<3*Math.PI-2*deltaLambda)))
|| (sigma>-Math.PI && sigma<0 &&
   ((lamb12>sigma-delta+Math.PI+2*deltaLambda &&
        lamb12<Math.PI+2*deltaLambda)
   ||(lamb31>delta+Math.PI+2*deltaLambda &&
        lamb31<Math.PI+2*deltaLambda)
   ||(lamb12+lamb31>Math.PI+2*deltaLambda &&
        lamb12+lamb31<2*Math.PI-2*deltaLambda)))
|| sigma==Math.PI)
        f = 1;

//SE f=1 ENTÃO PODE HAVER TG

if(f==1)
        {
        l23 = v23.length();
        xM23 = x2 + (v23.x)/2;
        yM23 = y2 + (v23.y)/2;

        lambdaPi = lamb12+lamb31-2*deltaLambda;
        R = Math.abs(l23/(2*Math.sin(lambdaPi)));
        lCM23 = l23/(2*Math.tan(-(lambdaPi)));
        xC = xM23 - lCM23*(v23.y)/l23;
        yC = yM23 + lCM23*(v23.x)/l23;
        lC1 = Math.sqrt(Math.pow(xC-x1, 2)+Math.pow(yC-y1, 2));
        psi = Math.asin(R/lC1);
        lCM = R * Math.sin(psi);
        lTM = R * Math.cos(psi);
        uC1x = (x1 - xC)/lC1;
        uC1y = (y1 - yC)/lC1;
        uMTAx = -uC1y;
        uMTAy = uC1x;
        xM = xC + lCM*uC1x;
        yM = yC + lCM*uC1y;

        xTA = xM + lTM*uMTAx;
        yTA = yM + lTM*uMTAy;

        xTB = xM - lTM*uMTAx;
        yTB = yM - lTM*uMTAy;


        Vector3d vTAB1a = new Vector3d(x1 - xTA, y1 - yTA, 0);
        Vector3d vTAB2a = new Vector3d(x2 - xTA, y2 - yTA, 0);

        lambda12TA = vTAB1a.angle(vTAB2a);
        Vector3d vTAB1aTAB2a = new Vector3d();
```

```
vTAB1aTAB2a.cross(vTAB1a,vTAB2a);
if(vTAB1aTAB2a.z < 0)
        lambda12TA = Math.PI*2 - lambda12TA;

if(lambda12TA <= lamb12 &&
                lambda12TA >= lamb12 - 2*deltaLambda)
        {
        Vector3d vB1TA = new Vector3d(xTA-x1, yTA-y1, 0);
        deltaTauTA = vB1TA.angle(vB1P0);
        if (deltaTauTA > deltaTauMax)
                deltaTauMax = deltaTauTA;
        }

Vector3d vTBB1a = new Vector3d(x1 - xTB, y1 - yTB, 0);
Vector3d vTBB2a = new Vector3d(x2 - xTB, y2 - yTB, 0);

lambda12TB = vTBB1a.angle(vTBB2a);
Vector3d vTBB1aTBB2a = new Vector3d();
vTBB1aTBB2a.cross(vTBB1a,vTBB2a);
if(vTBB1aTBB2a.z < 0)
        lambda12TB = Math.PI*2 - lambda12TB;

if(lambda12TB <= lamb12 &&
                lambda12TB >= lamb12 - 2*deltaLambda)
        {
        Vector3d vB1TB = new Vector3d(xTB-x1, yTB-y1, 0);
        deltaTauTB = vB1TB.angle(vB1P0);
        if (deltaTauTB > deltaTauMax)
                deltaTauMax = deltaTauTB;
        }


lambdaPi = lamb12+lamb31+2*deltaLambda;
R = Math.abs(l23/(2*Math.sin(lambdaPi)));
lCM23 = l23/(2*Math.tan(-(lambdaPi)));
xC = xM23 - lCM23*(v23.y)/l23;
yC = yM23 + lCM23*(v23.x)/l23;
lC1 = Math.sqrt(Math.pow(xC-x1, 2)+Math.pow(yC-y1, 2));
psi = Math.asin(R/lC1);
lCM = R * Math.sin(psi);
lTM = R * Math.cos(psi);
uC1x = (x1 - xC)/lC1;
uC1y = (y1 - yC)/lC1;
uMTAx = -uC1y;
uMTAy = uC1x;
xM = xC + lCM*uC1x;
yM = yC + lCM*uC1y;

xTA = xM + lTM*uMTAx;
yTA = yM + lTM*uMTAy;

xTB = xM - lTM*uMTAx;
yTB = yM - lTM*uMTAy;


Vector3d vTAB1b = new Vector3d(x1 - xTA, y1 - yTA, 0);
Vector3d vTAB2b = new Vector3d(x2 - xTA, y2 - yTA, 0);

lambda12TA = vTAB1b.angle(vTAB2b);
Vector3d vTAB1bTAB2b = new Vector3d();
vTAB1bTAB2b.cross(vTAB1b,vTAB2b);
if(vTAB1bTAB2b.z < 0)
        lambda12TA = Math.PI*2 - lambda12TA;

if(lambda12TA <= lamb12 &&
                lambda12TA >= lamb12 - 2*deltaLambda)
        {
        Vector3d vB1TA = new Vector3d(xTA-x1, yTA-y1, 0);
        deltaTauTA = vB1TA.angle(vB1P0);
        if (deltaTauTA > deltaTauMax)
                deltaTauMax = deltaTauTA;
        }

Vector3d vTBB1b = new Vector3d(x1 - xTB, y1 - yTB, 0);
Vector3d vTBB2b = new Vector3d(x2 - xTB, y2 - yTB, 0);

lambda12TB = vTBB1b.angle(vTBB2b);
Vector3d vTBB1bTBB2b = new Vector3d();
vTBB1bTBB2b.cross(vTBB1b,vTBB2b);
if(vTBB1bTBB2b.z < 0)
        lambda12TB = Math.PI*2 - lambda12TB;

if(lambda12TB <= lamb12 &&
                lambda12TB >= lamb12 - 2*deltaLambda)
        {
        Vector3d vB1TB = new Vector3d(xTB-x1, yTB-y1, 0);
        deltaTauTB = vB1TB.angle(vB1P0);
        if (deltaTauTB > deltaTauMax)
```

```
                                        deltaTauMax = deltaTauTB;
                                }
                        }


                        // CÁLCULO DE ERROS ---------------------------------------------


                        // ERRO DE POSIÇÃO

                        Vector2d vPos = new Vector2d(x, y);
                        Vector2d vErroPos = new Vector2d(xR[0]-x, yR[0]-y);
                                erroPos = vErroPos.length();

                        erroPosEstimado = distMax;

                        deltaErroPos = erroPos - erroPosEstimado;

                        // REALÇAR ERRO ESTIMADO INSUFICIENTE
                        if (deltaErroPos > 0)
                                deltaErroPos = erroPosMax;



                        // ERRO DE ORIENTAÇÃO

                        erroOr = Math.toDegrees(Math.abs(tetaR-teta));
                        if (erroOr > 180)
                                erroOr = 360 - erroOr;

                        erroOrEstimado = Math.toDegrees(deltaTauMax+deltaLambda);

                        deltaErroOr = erroOr - erroOrEstimado;

                        // REALÇAR ERRO ESTIMADO INSUFICIENTE
                        if (deltaErroOr > 0)
                                deltaErroOr = erroOrMax;

                }

                // SAÍDAS -------------------------------------------------------------

                // X e Y
                ps1.print(x + "\t");
                ps2.print(y + "\t");

                // ERROPOS
                if(erroPos > erroPosMax)
                        erroPos = erroPosMax;
                ps3.print(erroPos + "\t");

                // ERROOR
                if(erroOr > erroOrMax)
                        erroOr = erroOrMax;
                ps4.print(erroOr + "\t");

                // ERROPOSESTIMADO
                if(erroPosEstimado > erroPosEstimadoMax)
                        erroPosEstimado = erroPosEstimadoMax;
                ps5.print(erroPosEstimado + "\t");

                // DELTAERROPOS
                if(deltaErroPos > deltaErroPosMax)
                        deltaErroPos = deltaErroPosMax;
                if(deltaErroPos < - deltaErroPosMax)
                        deltaErroPos = - deltaErroPosMax;
                ps6.print(deltaErroPos + "\t");

                // ERROORSESTIMADO
                if(erroOrEstimado > erroOrEstimadoMax)
                        erroOrEstimado = erroOrEstimadoMax;
                ps7.print(erroOrEstimado + "\t");

                // DELTAERROOR
                if(deltaErroOr > deltaErroOrMax)
                        deltaErroOr = deltaErroOrMax;
                if(deltaErroOr < - deltaErroOrMax)
                        deltaErroOr = - deltaErroOrMax;
                ps8.print(deltaErroOr + "\t");

                //-----------------------------------------------------------------------

                x += dx;
        }
        ps1.println("");
        ps2.println("");
        ps3.println("");
        ps4.println("");
```

```
                ps5.println("");
                ps6.println("");
                ps7.println("");
                ps8.println("");

                y += dy;
        }
        ps1.close();
        ps2.close();
        ps3.close();
        ps4.close();
        ps5.close();
        ps6.close();
        ps7.close();
        ps8.close();
        ps9.close();
        }
}
```

I.4 Código Fonte do Programa Usado no Quarto Conjunto de Testes

```
import java.io.*;
import javax.vecmath.*;
class Ggt4
{
public static void main (String[] arguments)throws IOException
        {
        double x1, y1, x2, y2, x3, y3, xMIN, yMIN, xMAX, yMAX, dx, dy, casDec;
        double x, y, teta, l12, l23, l31, lambda1, lambda2, lambda3, lamb12, lamb23, lamb31,
                                                                                    deltaLambda;
        double xR[]=new double [7];
        double yR[]=new double [7];
        double tetaR;
        double fi, sigma, gama, delta, num, den, tau, l1;
        double erroPosEstimado, erroOrEstimado;
        double dist30, dist40, dist50, dist60, dist10, dist20, dist70, dist80;
        double dist12, dist23, dist34, dist45, dist56, dist61, distMax, dmax;
        double deltaTau10, deltaTau20, deltaTau30, deltaTau40, deltaTau50, deltaTau60;
        double deltaTauTA, deltaTauTB, deltaTauMax;
        double ang12, ang23, ang34, ang45, ang56, ang61;
        double f;
        double w, rT, xCL,yCL, distCLB1, distCLB2, distCLB3;
        double lambdaPi, psi, R, lCM23, lCM, lC1, lTM, uC1x, uC1y, uMTAx, uMTAy;
        double xC, yC, xM23, yM23, xM, yM, xTA, yTA, xTB, yTB, lambda12TA, lambda12TB;
        double contTotal, contLocImpossivel, contTgPossivel, contTg, contEfectivo, aux;
        long tempoInicial, tempoFinal, tempoTotal;
        double tempoMedioIteracao, tempoMedioSimulacao;
        double pi2 = 2 * Math.PI;
        double pi3 = 3 * Math.PI;
        double inexist = Math.acos(2);


        // Parâmetros-------------------------------------------------------------------------------

        //COORDENADAS DAS BALIZAS
        x1 = 75;
        y1 = 75;
        x2 = 25;
        y2 = 60;
        x3 = 55;
        y3 = 25;


        //RECINTO DE NAVEGAÇÃO
        xMIN = 10.0;
        yMIN = 90.0;
        xMAX = 11.0;
        yMAX = 91.0;

        //INCREMENTOS
        dx = 0.001;
        dy = 0.001;

        casDec=0;
        deltaLambda = Math.toRadians(0.5);
        double deltaLambda2 = 2 * deltaLambda;

        // ----------------------------------------------------------------------------------------
```

```
Vector3d vHoriz = new Vector3d(1,0,0);

contTotal = 0;
contLocImpossivel = 0;
contTgPossivel = 0;
contTg = 0;
contEfectivo = 0;
tempoInicial = System.currentTimeMillis();

Vector3d vR0= new Vector3d();

Vector3d v12= new Vector3d();
Vector3d v31= new Vector3d();

Vector3d v1= new Vector3d(-1,0,0);

Vector3d v21= new Vector3d();
Vector3d v23= new Vector3d();

Vector3d vP0P1 = new Vector3d();
Vector3d vP0P2 = new Vector3d();
Vector3d vP0P3 = new Vector3d();
Vector3d vP0P4 = new Vector3d();
Vector3d vP0P5 = new Vector3d();
Vector3d vP0P6 = new Vector3d();

Vector3d sub12 = new Vector3d();
Vector3d sub23 = new Vector3d();
Vector3d sub34 = new Vector3d();
Vector3d sub45 = new Vector3d();
Vector3d sub56 = new Vector3d();
Vector3d sub61 = new Vector3d();

Vector3d vB1P0 = new Vector3d();
Vector3d vB1P1 = new Vector3d();
Vector3d vB1P2 = new Vector3d();
Vector3d vB1P3 = new Vector3d();
Vector3d vB1P4 = new Vector3d();
Vector3d vB1P5 = new Vector3d();
Vector3d vB1P6 = new Vector3d();

Vector3d v112 = new Vector3d();
Vector3d v3112 = new Vector3d();

Vector3d vTAB1a = new Vector3d();
Vector3d vTAB2a = new Vector3d();

Vector3d vTBB1a = new Vector3d();
Vector3d vTBB2a = new Vector3d();

Vector3d vTAB1aTAB2a = new Vector3d();
Vector3d vTBB1aTBB2a = new Vector3d();

Vector3d vTAB1b = new Vector3d();
Vector3d vTAB2b = new Vector3d();

Vector3d vTBB1b = new Vector3d();
Vector3d vTBB2b = new Vector3d();

Vector3d vTAB1bTAB2b = new Vector3d();
Vector3d vTBB1bTBB2b = new Vector3d();

Vector3d vB1TA = new Vector3d();
Vector3d vB1TB = new Vector3d();

Vector3d vR1 = new Vector3d();
Vector3d vR2 = new Vector3d();
Vector3d vR3 = new Vector3d();

Vector3d vR0R1 = new Vector3d();
Vector3d vR0R2 = new Vector3d();
Vector3d vR0R3 = new Vector3d();


y = yMIN;
while(y<=yMAX)
{
        x = xMIN;
        while(x<=xMAX)
        {
                contTotal++;

                vR0.x= Math.pow(-1, Math.rint(Math.random()*10))* Math.random()*10;
                vR0.y= Math.pow(-1, Math.rint(Math.random()*10))* Math.random()*10;
                vR0.z= 0;

                if(vR0.length() == 0)
                        vR0.x = 1;
```

```
// CÁLCULO DE lambda12 E lambda31----------------------------------------

vR1.x = x1-x;
vR1.y = y1-y;
vR1.z = 0;

vR2.x = x2-x;
vR2.y = y2-y;
vR2.z = 0;

vR3.x = x3-x;
vR3.y = y3-y;
vR3.z = 0;


lambda1 = vR0.angle(vR1);
vR0R1.cross(vR0,vR1);
if(vR0R1.z < 0)
        lambda1 = pi2 - lambda1;

lambda2 = vR0.angle(vR2);
vR0R2.cross(vR0,vR2);
if(vR0R2.z < 0)
        lambda2 = pi2 - lambda2;

lambda3 = vR0.angle(vR3);
vR0R3.cross(vR0,vR3);
if(vR0R3.z < 0)
        lambda3 = pi2 - lambda3;


//INTRODUZIR ERROS

lambda1 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                *Math.toDegrees(lambda1))/Math.pow(10,casDec));

lambda2 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                *Math.toDegrees(lambda2))/Math.pow(10,casDec));

lambda3 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                *Math.toDegrees(lambda3))/Math.pow(10,casDec));


lamb12 = lambda2 - lambda1;
if(lambda2 < lambda1)
        lamb12 += pi2;

lamb31 = lambda1 - lambda3;
if(lambda1 < lambda3)
        lamb31 += pi2;

lamb23 = lambda3 - lambda2;
if(lambda3 < lambda2)
        lamb23 += pi2;



// CÁLCULO DE l12 E l31 ------------------------------------------------

v12.x= x2-x1;
v12.y= y2-y1;
v12.z= 0;

v31.x= x1-x3;
v31.y= y1-y3;
v31.z= 0;

l12 = v12.length();
l31 = v31.length();


// CÁLCULO DE fi -------------------------------------------------------
fi = v1.angle(v12);

v112.cross(v1,v12);
if(v112.z < 0)
        fi = -fi;


// CÁLCULO DE sigma ----------------------------------------------------

sigma = v31.angle(v12);

v3112.cross(v31,v12);
if(v3112.z < 0)
        sigma = -sigma;
```

```
// CÁLCULO DE delta ----------------------------------------------------

v21.x= x1-x2;
v21.y= y1-y2;
v21.z= 0;

v23.x= x3-x2;
v23.y= y3-y2;
v23.z= 0;

delta = v21.angle(v23);
if(sigma < 0)
        delta = -delta;


if(Math.abs(lamb12+lamb31-sigma-Math.PI)<=deltaLambda2
        || Math.abs(lamb12+lamb31-sigma-pi3)<=deltaLambda2
        || (sigma<0 && (Math.abs(lamb12-sigma+delta-pi2)<=deltaLambda2
                        || Math.abs(lamb31-delta-pi2)<=deltaLambda2))
        || (sigma>=0 && (Math.abs(lamb12-sigma+delta)<=deltaLambda2
                        || Math.abs(lamb31-delta)<=deltaLambda2)))
{
contLocImpossivel++;
erroPosEstimado = inexist;
erroOrEstimado = inexist;
}

else
{
contEfectivo++;

// CÁLCULO DA POSIÇÃO E DA ORIENTAÇÃO DO ROBÔ ----------------------------

double lambda12[]={lamb12, lamb12-deltaLambda2, lamb12-deltaLambda2,
                lamb12, lamb12+deltaLambda2, lamb12+deltaLambda2, lamb12};
double lambda31[]={lamb31, lamb31, lamb31+deltaLambda2,
                lamb31+deltaLambda2, lamb31, lamb31-deltaLambda2,
                lamb31-deltaLambda2};


        gama = sigma - lambda31[0];

        num = Math.sin(lambda12[0])*(l12*Math.sin(lambda31[0])
                -l31*Math.sin(gama));
        den = l31*Math.sin(lambda12[0])*Math.cos(gama)
                -l12*Math.cos(lambda12[0])*Math.sin(lambda31[0]);
        tau = Math.atan(num/den);

        if((lambda12[0] < Math.PI) && (tau < 0))
                tau = tau + Math.PI;
        if((lambda12[0] > Math.PI) && (tau > 0))
                tau = tau - Math.PI;

        if(tau == 0 && ((sigma<0 && lambda31[0]<Math.PI)
                        || (sigma>0 && lambda31[0]>Math.PI)))
                tau = Math.PI;

        if(Math.abs(Math.sin(lambda12[0])) >
                                Math.abs(Math.sin(lambda31[0])))
                l1 = l12*Math.sin(tau + lambda12[0])
                                        /Math.sin(lambda12[0]);
        else
                l1 = l31*Math.sin(tau + sigma - lambda31[0])
                                        /Math.sin(lambda31[0]);


        xR[0] = x1-l1*Math.cos(fi+tau);
        yR[0] = y1-l1*Math.sin(fi+tau);

        tetaR = fi+tau-lambda1;

        if(tetaR <= -Math.PI)
                tetaR = tetaR + pi2;
        if(tetaR > Math.PI)
                tetaR = tetaR - pi2;


// CÁLCULO DAS COORDENADAS DOS VÉRTICES DA SUP. DE INCERTEZA DE POSIÇÃO ---

for (int i=1; i<7; i++)
        {
        gama = sigma - lambda31[i];

        num = Math.sin(lambda12[i])*(l12*Math.sin(lambda31[i])
                -l31*Math.sin(gama));
        den = l31*Math.sin(lambda12[i])*Math.cos(gama)
                -l12*Math.cos(lambda12[i])*Math.sin(lambda31[i]);
        tau = Math.atan(num/den);
```

```
if((lambda12[i] < Math.PI) && (tau < 0))
        tau = tau + Math.PI;
if((lambda12[i] > Math.PI) && (tau > 0))
        tau = tau - Math.PI;

if(tau == 0 && ((sigma<0 && lambda31[i]<Math.PI)
                || (sigma>0 && lambda31[i]>Math.PI)))
        tau = Math.PI;

if(Math.abs(Math.sin(lambda12[i])) >
                        Math.abs(Math.sin(lambda31[i])))
        l1 = l12*Math.sin(tau + lambda12[i])
                                /Math.sin(lambda12[i]);
else
        l1 = l31*Math.sin(tau + sigma - lambda31[i])
                                /Math.sin(lambda31[i]);

xR[i] = x1-l1*Math.cos(fi+tau);
yR[i] = y1-l1*Math.sin(fi+tau);
}



//CÁLCULO DE distMax ------------------------------------------

vP0P1.x = xR[1]-xR[0];
vP0P1.y = yR[1]-yR[0];
vP0P1.z = 0;

vP0P2.x = xR[2]-xR[0];
vP0P2.y = yR[2]-yR[0];
vP0P2.z = 0;

vP0P3.x = xR[3]-xR[0];
vP0P3.y = yR[3]-yR[0];
vP0P3.z = 0;

vP0P4.x = xR[4]-xR[0];
vP0P4.y = yR[4]-yR[0];
vP0P4.z = 0;

vP0P5.x = xR[5]-xR[0];
vP0P5.y = yR[5]-yR[0];
vP0P5.z = 0;

vP0P6.x = xR[6]-xR[0];
vP0P6.y = yR[6]-yR[0];
vP0P6.z = 0;

dist10 = vP0P1.length();
dist20 = vP0P2.length();
dist30 = vP0P3.length();
dist40 = vP0P4.length();
dist50 = vP0P5.length();
dist60 = vP0P6.length();

sub12.sub(vP0P2,vP0P1);
dist12 = sub12.length();

sub23.sub(vP0P3,vP0P2);
dist23 = sub23.length();

sub34.sub(vP0P4,vP0P3);
dist34 = sub34.length();

sub45.sub(vP0P5,vP0P4);
dist45 = sub45.length();

sub56.sub(vP0P6,vP0P5);
dist56 = sub56.length();

sub61.sub(vP0P1,vP0P6);
dist61 = sub61.length();

ang12 = vP0P1.angle(vP0P2);
ang23 = vP0P2.angle(vP0P3);
ang34 = vP0P3.angle(vP0P4);
ang45 = vP0P4.angle(vP0P5);
ang56 = vP0P5.angle(vP0P6);
ang61 = vP0P6.angle(vP0P1);

distMax = 0;
if (dist10 > distMax)
        distMax = dist10;
if (dist20 > distMax)
        distMax = dist20;
if (dist30 > distMax)
        distMax = dist30;
```

```
if (dist40 > distMax)
        distMax = dist40;
if (dist50 > distMax)
        distMax = dist50;
if (dist60 > distMax)
        distMax = dist60;


if (dist10 > dist20)
        {
        w = Math.asin(Math.sin(ang12)*dist20/dist12);
        rT = (dist12/2)/Math.cos(w);
        double quo = rT/dist10;
        xCL = xR[1] - (quo) * (xR[1] - xR[0]);
        yCL = yR[1] - (quo) * (yR[1] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang12)*dist10/dist12);
        rT = (dist12/2)/Math.cos(w);
        double quo = rT/dist20;
        xCL = xR[2] - (quo) * (xR[2] - xR[0]);
        yCL = yR[2] - (quo) * (yR[2] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));

if (distCLB1 < rT || distCLB2 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)
                                +Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist20 > dist30)
        {
        w = Math.asin(Math.sin(ang23)*dist30/dist23);
        rT = (dist23/2)/Math.cos(w);
        double quo = rT/dist20;
        xCL = xR[2] - (quo) * (xR[2] - xR[0]);
        yCL = yR[2] - (quo) * (yR[2] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang23)*dist20/dist23);
        rT = (dist23/2)/Math.cos(w);
        double quo = rT/dist30;
        xCL = xR[3] - (quo) * (xR[3] - xR[0]);
        yCL = yR[3] - (quo) * (yR[3] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));

if (distCLB1 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)
                                +Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist30 > dist40)
        {
        w = Math.asin(Math.sin(ang34)*dist40/dist34);
        rT = (dist34/2)/Math.cos(w);
        double quo = rT/dist30;
        xCL = xR[3] - (quo) * (xR[3] - xR[0]);
        yCL = yR[3] - (quo) * (yR[3] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang34)*dist30/dist34);
        rT = (dist34/2)/Math.cos(w);
        double quo = rT/dist40;
        xCL = xR[4] - (quo) * (xR[4] - xR[0]);
        yCL = yR[4] - (quo) * (yR[4] - yR[0]);
        }

distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));

if (distCLB2 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)
                                +Math.pow(yCL-yR[0], 2)) + rT;
```

```
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist40 > dist50)
        {
        w = Math.asin(Math.sin(ang45)*dist50/dist45);
        rT = (dist45/2)/Math.cos(w);
        double quo = rT/dist40;
        xCL = xR[4] - (quo) * (xR[4] - xR[0]);
        yCL = yR[4] - (quo) * (yR[4] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang45)*dist40/dist45);
        rT = (dist45/2)/Math.cos(w);
        double quo = rT/dist50;
        xCL = xR[5] - (quo) * (xR[5] - xR[0]);
        yCL = yR[5] - (quo) * (yR[5] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));

if (distCLB1 < rT || distCLB2 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)
                                +Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist50 > dist60)
        {
        w = Math.asin(Math.sin(ang56)*dist60/dist56);
        rT = (dist56/2)/Math.cos(w);
        double quo = rT/dist50;
        xCL = xR[5] - (quo) * (xR[5] - xR[0]);
        yCL = yR[5] - (quo) * (yR[5] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang56)*dist50/dist56);
        rT = (dist56/2)/Math.cos(w);
        double quo = rT/dist60;
        xCL = xR[6] - (quo) * (xR[6] - xR[0]);
        yCL = yR[6] - (quo) * (yR[6] - yR[0]);
        }

distCLB1 = Math.sqrt(Math.pow(xCL - x1, 2)+Math.pow(yCL - y1, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));

if (distCLB1 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)
                                +Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }

if (dist60 > dist10)
        {
        w = Math.asin(Math.sin(ang61)*dist10/dist61);
        rT = (dist61/2)/Math.cos(w);
        double quo = rT/dist60;
        xCL = xR[1] - (quo) * (xR[1] - xR[0]);
        yCL = yR[1] - (quo) * (yR[1] - yR[0]);
        }
else
        {
        w = Math.asin(Math.sin(ang61)*dist60/dist61);
        rT = (dist61/2)/Math.cos(w);
        double quo = rT/dist10;
        xCL = xR[1] - (quo) * (xR[1] - xR[0]);
        yCL = yR[1] - (quo) * (yR[1] - yR[0]);
        }

distCLB2 = Math.sqrt(Math.pow(xCL - x2, 2)+Math.pow(yCL - y2, 2));
distCLB3 = Math.sqrt(Math.pow(xCL - x3, 2)+Math.pow(yCL - y3, 2));

if (distCLB2 < rT || distCLB3 < rT)
        {
        dmax = Math.sqrt(Math.pow(xCL-xR[0], 2)
                                +Math.pow(yCL-yR[0], 2)) + rT;
        if (distMax < dmax)
                distMax = dmax;
        }
```

```
//CÁLCULO DE deltaTauMax ----------------------------------------

vB1P0.x = xR[0]-x1;
vB1P0.y = yR[0]-y1;
vB1P0.z = 0;

vB1P1.x = xR[1]-x1;
vB1P1.y = yR[1]-y1;
vB1P1.z = 0;

vB1P2.x = xR[2]-x1;
vB1P2.y = yR[2]-y1;
vB1P2.z = 0;

vB1P3.x = xR[3]-x1;
vB1P3.y = yR[3]-y1;
vB1P3.z = 0;

vB1P4.x = xR[4]-x1;
vB1P4.y = yR[4]-y1;
vB1P4.z = 0;

vB1P5.x = xR[5]-x1;
vB1P5.y = yR[5]-y1;
vB1P5.z = 0;

vB1P6.x = xR[6]-x1;
vB1P6.y = yR[6]-y1;
vB1P6.z = 0;


deltaTau10 = vB1P1.angle(vB1P0);
deltaTau20 = vB1P2.angle(vB1P0);
deltaTau30 = vB1P3.angle(vB1P0);
deltaTau40 = vB1P4.angle(vB1P0);
deltaTau50 = vB1P5.angle(vB1P0);
deltaTau60 = vB1P6.angle(vB1P0);

deltaTauMax = 0;

if (deltaTau10 > deltaTauMax)
        deltaTauMax = deltaTau10;
if (deltaTau20 > deltaTauMax)
        deltaTauMax = deltaTau20;
if (deltaTau30 > deltaTauMax)
        deltaTauMax = deltaTau30;
if (deltaTau40 > deltaTauMax)
        deltaTauMax = deltaTau40;
if (deltaTau50 > deltaTauMax)
        deltaTauMax = deltaTau50;
if (deltaTau60 > deltaTauMax)
        deltaTauMax = deltaTau60;

aux = deltaTauMax;


//SE f=0 ENTÃO NÃO PODE HAVER TG

f = 0;
if((sigma>0 && sigma<Math.PI &&
   ((lamb12>Math.PI-deltaLambda2 &&
        lamb12<sigma-delta+Math.PI-deltaLambda2)
   ||(lamb31>Math.PI-deltaLambda2 &&
        lamb31<delta+Math.PI-deltaLambda2)
   ||(lamb12+lamb31>pi2+deltaLambda2 &&
        lamb12+lamb31<pi3-deltaLambda2)))
|| (sigma>-Math.PI && sigma<0 &&
   ((lamb12>sigma-delta+Math.PI+deltaLambda2 &&
        lamb12<Math.PI+deltaLambda2)
   ||(lamb31>delta+Math.PI+deltaLambda2 &&
        lamb31<Math.PI+deltaLambda2)
   ||(lamb12+lamb31>Math.PI+deltaLambda2 &&
        lamb12+lamb31<pi2-deltaLambda2)))
|| sigma==Math.PI)
        f = 1;

//SE f=1 ENTÃO PODE HAVER TG

if(f==1)
        {
        contTgPossivel++;
        l23 = v23.length();
        xM23 = x2 + (v23.x)/2;
        yM23 = y2 + (v23.y)/2;

        lambdaPi = lamb12+lamb31-deltaLambda2;
        R = Math.abs(l23/(2*Math.sin(lambdaPi)));
        lCM23 = l23/(2*Math.tan(-(lambdaPi)));
```

```
xC = xM23 - lCM23*(v23.y)/l23;
yC = yM23 + lCM23*(v23.x)/l23;
lC1 = Math.sqrt(Math.pow(xC-x1, 2)+Math.pow(yC-y1, 2));
psi = Math.asin(R/lC1);
lCM = R * Math.sin(psi);
lTM = R * Math.cos(psi);
uC1x = (x1 - xC)/lC1;
uC1y = (y1 - yC)/lC1;
uMTAx = -uC1y;
uMTAy = uC1x;
xM = xC + lCM*uC1x;
yM = yC + lCM*uC1y;

xTA = xM + lTM*uMTAx;
yTA = yM + lTM*uMTAy;

xTB = xM - lTM*uMTAx;
yTB = yM - lTM*uMTAy;



vTAB1a.x = x1 - xTA;
vTAB1a.y = y1 - yTA;
vTAB1a.z = 0;

vTAB2a.x = x2 - xTA;
vTAB2a.y = y2 - yTA;
vTAB2a.z = 0;

lambda12TA = vTAB1a.angle(vTAB2a);
vTAB1aTAB2a.cross(vTAB1a,vTAB2a);
if(vTAB1aTAB2a.z < 0)
        lambda12TA = pi2 - lambda12TA;

if(lambda12TA <= lamb12 &&
                lambda12TA >= lamb12 - deltaLambda2)
        {
        vB1TA.x = xTA - x1;
        vB1TA.y = yTA - y1;
        vB1TA.z = 0;
        deltaTauTA = vB1TA.angle(vB1P0);
        if (deltaTauTA > deltaTauMax)
                deltaTauMax = deltaTauTA;
        }

vTBB1a.x = x1 - xTB;
vTBB1a.y = y1 - yTB;
vTBB1a.z = 0;

vTBB2a.x = x2 - xTB;
vTBB2a.y = y2 - yTB;
vTBB2a.z = 0;


lambda12TB = vTBB1a.angle(vTBB2a);
vTBB1aTBB2a.cross(vTBB1a,vTBB2a);
if(vTBB1aTBB2a.z < 0)
        lambda12TB = pi2 - lambda12TB;

if(lambda12TB <= lamb12 &&
                lambda12TB >= lamb12 - deltaLambda2)
        {
        vB1TB.x = xTB - x1;
        vB1TB.y = yTB - y1;
        vB1TB.z = 0;
        deltaTauTB = vB1TB.angle(vB1P0);
        if (deltaTauTB > deltaTauMax)
                deltaTauMax = deltaTauTB;
        }


lambdaPi = lamb12+lamb31+deltaLambda2;
R = Math.abs(l23/(2*Math.sin(lambdaPi)));
lCM23 = l23/(2*Math.tan(-(lambdaPi)));
xC = xM23 - lCM23*(v23.y)/l23;
yC = yM23 + lCM23*(v23.x)/l23;
lC1 = Math.sqrt(Math.pow(xC-x1, 2)+Math.pow(yC-y1, 2));
psi = Math.asin(R/lC1);
lCM = R * Math.sin(psi);
lTM = R * Math.cos(psi);
uC1x = (x1 - xC)/lC1;
uC1y = (y1 - yC)/lC1;
uMTAx = -uC1y;
uMTAy = uC1x;
xM = xC + lCM*uC1x;
yM = yC + lCM*uC1y;
```

```
                                        xTA = xM + lTM*uMTAx;
                                        yTA = yM + lTM*uMTAy;

                                        xTB = xM - lTM*uMTAx;
                                        yTB = yM - lTM*uMTAy;


                                        vTAB1b.x = x1 - xTA;
                                        vTAB1b.y = y1 - yTA;
                                        vTAB1b.z = 0;

                                        vTAB2b.x = x2 - xTA;
                                        vTAB2b.y = y2 - yTA;
                                        vTAB2b.z = 0;

                                        lambda12TA = vTAB1b.angle(vTAB2b);
                                        vTAB1bTAB2b.cross(vTAB1b,vTAB2b);
                                        if(vTAB1bTAB2b.z < 0)
                                                lambda12TA = pi2 - lambda12TA;

                                        if(lambda12TA <= lamb12 &&
                                                        lambda12TA >= lamb12 - deltaLambda2)
                                                {
                                                vB1TA.x = xTA - x1;
                                                vB1TA.y = yTA - y1;
                                                vB1TA.z = 0;
                                                deltaTauTA = vB1TA.angle(vB1P0);
                                                if (deltaTauTA > deltaTauMax)
                                                        deltaTauMax = deltaTauTA;
                                                }

                                        vTBB1b.x = x1 - xTB;
                                        vTBB1b.y = y1 - yTB;
                                        vTBB1b.z = 0;

                                        vTBB2b.x = x2 - xTB;
                                        vTBB2b.y = y2 - yTB;
                                        vTBB2b.z = 0;

                                        lambda12TB = vTBB1b.angle(vTBB2b);
                                        vTBB1bTBB2b.cross(vTBB1b,vTBB2b);
                                        if(vTBB1bTBB2b.z < 0)
                                                lambda12TB = pi2 - lambda12TB;

                                        if(lambda12TB <= lamb12 &&
                                                        lambda12TB >= lamb12 - deltaLambda2)
                                                {
                                                vB1TB.x = xTB - x1;
                                                vB1TB.y = yTB - y1;
                                                vB1TB.z = 0;
                                                deltaTauTB = vB1TB.angle(vB1P0);
                                                if (deltaTauTB > deltaTauMax)
                                                        deltaTauMax = deltaTauTB;
                                                }

                                        if (aux!=deltaTauMax)
                                                contTg++;
                                        }


                                // CÁLCULO DE ERROS ---------------------------------------------

                                // ERRO DE POSIÇÃO

                                erroPosEstimado = distMax;


                                // ERRO DE ORIENTAÇÃO

                                erroOrEstimado = Math.toDegrees(deltaTauMax+deltaLambda);

                                // -------------------------------------------------------------

                        }
                        x += dx;
                }
                y += dy;
        }

        tempoFinal = System.currentTimeMillis();
        tempoTotal = tempoFinal - tempoInicial;

        tempoMedioIteracao = tempoTotal;
        tempoMedioIteracao = tempoMedioIteracao/contTotal;

        tempoMedioSimulacao = tempoTotal;
        tempoMedioSimulacao = tempoMedioSimulacao/contEfectivo;
```

```
  long tempoFinal2;
  long tempoInicial2;

  tempoInicial2= System.currentTimeMillis();


  y = yMIN;
  while(y<=yMAX)
  {
          x = xMIN;
          while(x<=xMAX)
          {
           vR0.x= Math.pow(-1, Math.rint(Math.random()*10))* Math.random()*10;
           vR0.y= Math.pow(-1, Math.rint(Math.random()*10))* Math.random()*10;
           vR0.z= 0;

                  if(vR0.length() == 0)
                          vR0.x = 1;


                  // CÁLCULO DE lambda12 E lambda31----------------------------------------

                  vR1.x = x1-x;
                  vR1.y = y1-y;
                  vR1.z = 0;

                  vR2.x = x2-x;
                  vR2.y = y2-y;
                  vR2.z = 0;

                  vR3.x = x3-x;
                  vR3.y = y3-y;
                  vR3.z = 0;


                  lambda1 = vR0.angle(vR1);
                  vR0R1.cross(vR0,vR1);
                  if(vR0R1.z < 0)
                          lambda1 = pi2 - lambda1;

                  lambda2 = vR0.angle(vR2);
                  vR0R2.cross(vR0,vR2);
                  if(vR0R2.z < 0)
                          lambda2 = pi2 - lambda2;

                  lambda3 = vR0.angle(vR3);
                  vR0R3.cross(vR0,vR3);
                  if(vR0R3.z < 0)
                          lambda3 = pi2 - lambda3;


                  //INTRODUZIR ERROS

                  lambda1 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                                  *Math.toDegrees(lambda1))/Math.pow(10,casDec));

                  lambda2 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                                  *Math.toDegrees(lambda2))/Math.pow(10,casDec));

                  lambda3 = Math.toRadians(Math.rint(Math.pow(10,casDec)
                                  *Math.toDegrees(lambda3))/Math.pow(10,casDec));


                  lamb12 = lambda2 - lambda1;
                  if(lambda2 < lambda1)
                          lamb12 += pi2;

                  lamb31 = lambda1 - lambda3;
                  if(lambda1 < lambda3)
                          lamb31 += pi2;

                  lamb23 = lambda3 - lambda2;
                  if(lambda3 < lambda2)
                          lamb23 += pi2;


           x += dx;
      }
      y += dy;
}


tempoFinal2 = System.currentTimeMillis();

long tempoTotal2 = tempoFinal2 - tempoInicial2;
double tempoMed = ((double)(tempoTotal - tempoTotal2))/((double)contTotal);
double tempoMed2 = ((double)(tempoTotal - tempoTotal2))/((double)contEfectivo);
```

```
        // SAÍDA ---------------------------------------------------------------------------------

        PrintStream ps1 = new PrintStream (new FileOutputStream("c:/jse/trabalho/result/GGT4.txt"));
        ps1.println("x1 = " + x1);
        ps1.println("y1 = " + y1);
        ps1.println("x2 = " + x2);
        ps1.println("y2 = " + y2);
        ps1.println("x3 = " + x3);
        ps1.println("y3 = " + y3);
        ps1.println("xMIN = " + xMIN);
        ps1.println("yMIN = " + yMIN);
        ps1.println("xMAX = " + xMAX);
        ps1.println("yMAX = " + yMAX);
        ps1.println("dx = " + dx);
        ps1.println("dy = " + dy);
        ps1.println("casDec = " + casDec);
        ps1.println("deltaLambda = " + deltaLambda + "rad = " + Math.toDegrees(deltaLambda) + "º");
        ps1.println("Número total de iterações = " + contTotal);
        ps1.println("Número de iterações em que não foi possível fazer a localização = "
                                                                + contLocImpossivel);
        ps1.println("Número de iterações em que se fez a localização = " + contEfectivo);
        ps1.println("Número de iterações em que se verificou se deltaTauMax não ocorre num vértice
                              da superfície de incerteza de posição = " + contTgPossivel);
        ps1.println("Número de iterações em que deltaTauMax não ocorreu num vértice da superfície
                                                        de incerteza de posição = " + contTg);
        ps1.println("Tempo total = " + tempoTotal + "ms");
        ps1.println("Tempo médio de uma iteração = " + tempoMedioIteracao + "ms");
        ps1.println("Tempo médio de uma iteração em que se faz a localização = "
                                                                + tempoMedioSimulacao + "ms");
        ps1.println("Tempo médio de uma iteração (real) = " + tempoMed + "ms");
        ps1.println("Tempo médio de uma iteração em que se faz a localização (real) = "
                                                                + tempoMed2+ "ms");


        ps1.close();

        //---------------------------------------------------------------------------------------

        }
}
```