



UNIVERSIDADE FEDERAL DE SANTA CATARINA

Centro Tecnológico, de Ciências Exatas e Educação
Departamento de Matemática

PIBIC
RELATÓRIO FINAL

Geometria de Distâncias e Álgebras Geométricas: novas perspectivas
geométricas, computacionais e aplicações

**Disposição de Robôs Móveis no espaço
Euclidiano 3D: uma aplicação de Geometria
de Distâncias**

Guilherme Philippi (guilherme.philippi@hotmail.com),

ORIENTADOR: Felipe Delfini Caetano Fidalgo (felipe.fidalgo@ufsc.br).

17 de agosto de 2020

Sumário

1	Introdução	3
2	Materiais e Métodos	4
2.1	Teoria de Grafos	4
2.1.1	Descoberta (Eureka!)	4
2.1.2	Definição	5
2.1.3	Outras Definições Importantes	7
2.1.4	Grafos Ponderados	9
2.2	Geometria de Distâncias Euclidianas	10
2.2.1	Como tudo Começou	10
2.2.2	O Problema Fundamental	12
2.2.3	Os Diferentes Problemas em DG	12
2.2.4	A Busca de uma Solução	15
2.2.5	Combinatória do DGP	16
3	Resultados e Discussão	22
3.1	Robótica Móvel	22
3.1.1	Sensores	23
3.1.2	Como medir distâncias	24
3.1.3	Sobre o conjunto de sensores	26
3.2	Medindo erros	26
3.3	Exemplares utilizados	27
3.4	Simulações	28
4	Considerações Finais	31
	Referências Bibliográficas	33
A	Métricas	37
B	Rigidez	38

Abstract

In this work, the application of the so called Distance Geometry Problem to the Sensor Location Problem was studied, as well as the necessary tools for its understanding, from Graph Theory to the characteristics of systems involving mobile robotics. An overview of Distance Geometry was presented, which enabled the correct definition of the problem and polynomial algorithms to solve it. The text ends with an analysis of computer simulations of the problem, using different geometries, as well as an algorithm to generate them.

Keywords: Distance Geometry, Mobile Robotics.

Resumo

Neste trabalho, estudou-se o assim chamado Problema de Geometria de Distâncias aplicado ao Problema de Localização de Sensores, bem como as ferramentas necessárias para sua compreensão, passando da teoria de grafos às características de sistemas envolvendo robótica móvel. Apresentou-se uma visão geral de Geometria de Distâncias, o que possibilitou a correta definição do problema e de algoritmos polinomiais para solucioná-lo. O texto se encerra com uma análise de simulações computacionais do problema, utilizando diferentes geometrias, bem como um algoritmo para gerá-las.

Palavras-chave: Geometria de Distâncias, Robótica Móvel.

Capítulo 1

Introdução

Percebe-se uma tendência de mercado em se adotar sistemas autônomos móveis para realizar funções que dependem de um grande número de máquinas, como o controle do estoque em um armazém, um sistema de entrega ou o cultivo e tratamento de grandes áreas de plantio [1]. O estudo da localização de robôs móveis é vasta na literatura [2, 3], o que demonstra a importância deste tema. Qualquer solução de engenharia móvel, seja autônoma ou não, precisa definir uma forma de localizar seu objeto de estudo. Por conta disso, no decorrer desse texto, estuda-se algumas soluções envolvendo Geometria de Distâncias para a obtenção de localização.

A Geometria de Distâncias (GD) é uma matéria de interesse relativamente recente da Ciência [4], disposta em uma vasta interseção entre diversas áreas do conhecimento. Em suma, ela se preocupa com os aspectos geométricos decorrentes dos objetos que estão dispostos em algum espaço e possuam uma relação de distância mensurável, advindas de origens tão variáveis quanto se queira. Como é o caso de estudo deste texto: distâncias entre robôs móveis com seu meio e, em particular, outros robôs móveis.

Para poder representar essas distâncias, apresenta-se na Seção 2.1 um estudo sobre a Teoria de Grafos — que demonstrou-se uma ferramenta importante para GD, em contraponto à utilização da representação matricial [4]. Em seguida, na Seção 2.2, enuncia-se o problema fundamental deste trabalho utilizando grafos e faz-se um estudo sobre algorítimos descritos na literatura para solucioná-lo.

Na última parte deste trabalho há uma introdução à robótica móvel, na Seção 3.1, onde apresenta-se um estudo sobre os sensores existentes para a obtenção de distâncias, seguidos por algumas simulações computacionais do problema, na Seção 3.4. Por fim, há discussões envolvendo esses resultados. O texto também contém dois apêndices, podendo serem usados como consulta, o que inclui algumas ferramentas matemáticas utilizadas.

Uma revisão bibliográfica completa pode ser encontrada no fim do documento, antes dos apêndices, na Página 33.

Capítulo 2

Materiais e Métodos

2.1 Teoria de Grafos

Esta seção tem como objetivo apresentar um breve resumo da *Teoria de Grafos*, tema amplamente estudado por diversos matemáticos e aplicado em diversas áreas do conhecimento como computação, engenharia e matemática [5].

2.1.1 Descoberta (Eureka!)

Costuma-se dizer que a teoria se iniciou em 1736, com base no artigo publicado por Leonhard Euler (1707 a 1783) sobre as 7 pontes de Königsberg [6, 5], representada na Figura 2.1. Conta a história que os moradores daquela região perguntavam-se sobre a possibilidade de atravessar todas as sete pontes do local sem ter que repetir alguma delas. Esse é um problema muito usado para introduzir o tema [7] — propõe-se o desafio de ligar todos os pontos de um desenho sem tirar o lápis do papel e sem passar duas vezes no mesmo ponto. Para o caso das pontes de Königsberg, Euler provou que era impossível fazer isso ao formular matematicamente o problema, dando origem a esta teoria.

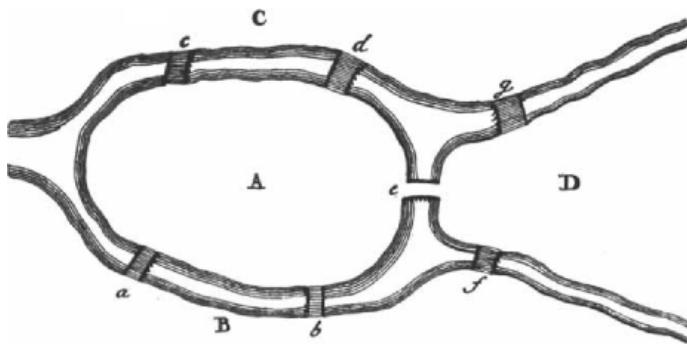


Figura 2.1: Ilustração original do problema [6].

A grande ideia de Euler foi abstrair o problema:vê-lo de uma forma elementar, como um conjunto de pontos conectados por curvas. Isso pode ser representado por um “gráfico”, conforme a Figura 2.2 — é daí a origem do termo em inglês “Graph”, que é tradução literal de “Gráfico”. Essa representação facilita a análise e a busca por uma solução. Com isso, Euler percebeu que só seria possível solucionar o

problema se houvessem exatamente nenhum ou apenas dois pontos conectados por um número ímpar de curvas (ou pontes) — o par de caminhos está associado com o ato de entrar e sair de um ponto [6]. Note que o caso de Koenigsberg, por tanto, não possui solução.

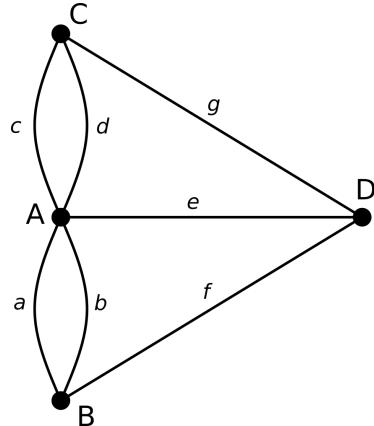


Figura 2.2: Grafo representando o caso da ponte de Koenigsberg.

Mas não podemos deixar todo o mérito com Euler. O conceito de grafo é muito intuitivo e foi proposto por diversas mentes brilhantes como forma de solucionar problemas que, em essência, são muito parecidos. Por exemplo, após Euler, a teoria foi redescoberta por Kirchhoff e Cayley [8]. Kirchhoff desenvolveu esse conceito por volta de 1847, enquanto solucionava sistemas de equações lineares que relacionavam as correntes que percorriam as malhas de um circuito elétrico [9]. Dez anos depois, em 1857, foi a vez de Cayley, que estudava diferentes estruturas em bioquímica formadas por carbonos (com quatro ligações químicas) e hidrogênios (com apenas uma ligação), onde conseguiu formular seu problema introduzindo o conceito de árvore em grafos [10].

Além dessas, muitas outras situações reais podem ser convenientemente representadas por simples diagramas contendo um conjunto de pontos e um conjunto de relações entre pares desses pontos. Por exemplo, pode-se definir o conjunto $P = \{a, b, c\}$ das pessoas a, b e c e um conjunto $A = \{\{a, b\}, \{b, c\}\}$ como o conjunto de amizades entre essas pessoas — no caso, a é amigo de b , que é amigo de c , porém a não é amigo de c . Esta análise se torna muitíssimo útil quando se deseja estudar como uma informação se propaga em redes sociais.

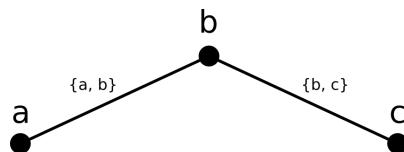


Figura 2.3: Grafo representando a relação entre as pessoas $\{a, b, c\}$.

2.1.2 Definição

Não há um forte consenso sobre as terminologias usadas pelos autores sobre grafos [8]. Essa confusão se deve tanto pela sua vasta disseminação em diversas áreas

como pela enorme abstração que ela carrega. Cayley poderia chamar as relações entre pontos de ligações químicas enquanto Kirchhoff chamaria de curto-circuitos. No que se segue, há aqui um apanhado de definições sobre a teoria de grafos. Mas não sobre toda ela. Essa é uma grande área da matemática e não cabe abordá-la completamente nesse texto. Trata-se apenas do essencial para que o leitor possa progredir sem ter que consultar uma bibliografia complementar sobre grafos.

O ideal é começar pela definição geral.

Definição: Um *grafo* G é uma tripla da forma (V_G, E_G, ψ_G) , composta por um *conjunto de vértices* V_G , um *conjunto de arestas* E_G e uma *função de incidência* ψ_G que, por sua vez, associa a cada elemento de E_G um par não ordenado de elementos (nem sempre distintos) de V_G .

Nesse texto, porém, abstraiu-se a função de incidência ψ_G pois entende-se que o conjunto de arestas E_G é tal que, se $e \in E_G$, então $e = \{a, b\}$ onde $a, b \in V_G$. Fica implícita, por tanto, a associação dos elementos de V_G e E_G .

Aos elementos dos conjuntos V_G e E_G , refere-se-os por *vértices* e *arestas*, respetivamente. Também, para uma aresta $e \in E_G$, onde $e = \{u, v\}$, diz-se que u e v são *vértices adjacentes*. Chama-se u e e de *incidentes*, assim como v e e . À quantidade de vértices adjacentes a v dá-se o nome *grau* de v . Para um vértice $v \in V_G$, define-se o *conjunto vizinhança* $N_G(v)$ como o conjunto de todos os vértices $u \in V_G$ adjacentes a v . Também, se duas arestas distintas e_1 e e_2 são incidentes com um vértice em comum, diz-se que e_1 e e_2 são *arestas adjacentes*.

Seja um grafo com m vértices e n arestas, dizer-se-á que este é um (m, n) *grafo*. Isto é, a Figura 2.3, para ilustrar, contém um $(3, 2)$ grafo onde os vértices a e b são adjacentes, assim como as arestas $\{a, b\}$ e $\{b, c\}$, porém, os vértices a e c não são. Define-se o $(1, 0)$ grafo como *trivial*.

Existem muitas variações de grafos [8]. Perceba que a definição de grafo permite *loops* (também chamado de *laço*, é uma aresta da forma $e = \{v, v\}$, ou seja, v é adjacente a si mesmo) e *múltiplas arestas* (mais do que uma aresta ligando os mesmos dois vértices). Grafos que não permitem múltiplas arestas ou loops são ditos *simples*. Grafos que permitem múltiplas arestas, mas não loops, são chamados de *multigrafos*. Caso também permitam os loops, os chamamos de *pseudografos*. Na Figura 2.2 (do problema das pontes de Koenigsberg) temos um multigrafo e na Figura 2.4 um pseudografo.

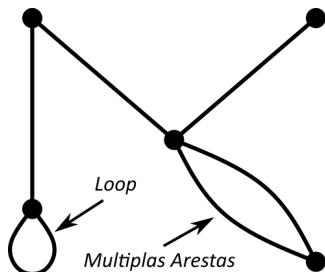


Figura 2.4: Exemplo de pseudografo contendo 5 vértices e 6 arestas.

Porém, para esse trabalho não interessa o estudo de multigrafos ou pseudografos. Por isso, adotou-se uma definição alternativa para grafos, visando restringir sua aplicação, como segue:

Definição: Um *grafo simple* G é uma dupla da forma (V_G, E_G) , composta por um conjunto não nulo e finito V_G e outro conjunto finito E_G de pares não ordenados de elementos **distintos** pertencentes a V_G .

2.1.3 Outras Definições Importantes

Diz-se que um (m, n) grafo G é *rotulado* quando pode-se distinguir seus m vértices ao nomeá-los — algo como v_1, v_2, \dots, v_m . Por exemplo, os grafos da Figura 2.5 são rotulados, enquanto o grafo da Figura 2.4 não é. Quando não é dito o contrário, considera-se todo grafo como rotulado.

Dois grafos $G = (V_G, E_G)$ e $H = (V_H, E_H)$ são ditos *isomorfos* (escreve-se $G \cong H$) quando existe uma correspondência biunívoca entre os conjuntos de vértices V_G e V_H que preserve suas adjacências. A Figura 2.5 ilustra essa situação, com a correspondência $v_i \longleftrightarrow v_i$.

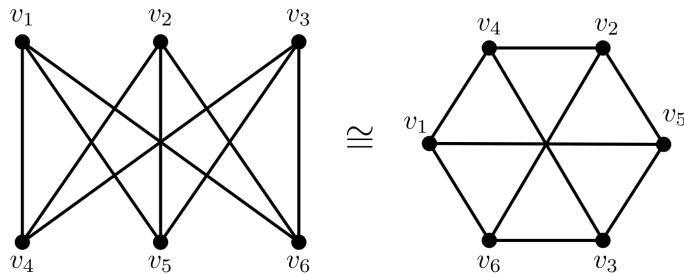


Figura 2.5: Diferentes representações isomórficas de um $(6, 9)$ grafo.

O isomorfismo é uma relação de equivalência em grafos. Fica claro que, por mais que seja útil, a representação gráfica de um grafo existe apenas como um apelo didático. A geometria formada pelos vértices é escolha de quem desenha. Vários são os casos em que problemas envolvendo grafos são facilmente solucionáveis apenas rearranjando a forma como se desenha — como o caso das pontes de Koenigsberg. A resposta salta aos olhos.

Subgrafos

Diz-se que o grafo $G_1 = (V_{G_1}, E_{G_1})$ é *subgrafo* de $G = (V_G, E_G)$ se $V_{G_1} \subset V_G$ e $E_{G_1} \subset E_G$. Se G_1 é subgrafo de G , então G é *supergrafo* de G_1 . Para qualquer $V \subset V_G$, existe um *subgrafo induzido* $\langle V \rangle$ definido por (V, E) , onde $E \subset E_G$ contém todas as arestas $(v_1, v_2) \in E_G$ tal que $v_1, v_2 \in V$. Fica claro que dois vértices em $\langle V \rangle$ são adjacentes se, e somente se, forem também adjacentes em G .

Pode-se *remover* um vértice v de um grafo $G = (V_G, E_G)$, que resulta no subgrafo induzido $G - v = \langle V_G \setminus \{v\} \rangle$. Da mesma forma, pode-se *remover* uma aresta e de um grafo $G = (V_G, E_G)$, resultando no grafo $G - e = (V_G, E_G \setminus \{e\})$.

Caminhos

Um *passeio* em G é uma sequência finita não nula $W = v_0e_1v_1e_2v_2 \dots e_kv_k$, onde seus termos são alternados entre vértices e arestas, tal que, para $1 \leq i \leq k$, antes e depois de e_i vem v_{i-1} e v_i , respectivamente. Diz-se que W é um passeio de v_0 para v_k , ou um (v_0, v_k) -passeio [5]. Os vértices v_0 e v_k são chamados origem e fim do passeio, respectivamente, e v_1, v_2, \dots, v_{k-1} são os vértices internos. O número k é o comprimento de W . Em um grafo simples, um passeio $v_0e_1v_1e_2v_2 \dots e_kv_k$ é determinado suficientemente pela sequência dos vértices que o constitui $v_0v_1v_2 \dots v_k$.

Se $W = v_0v_1 \dots v_k$ e $W' = v_kv_{k+1} \dots v_l$ são passeios, o passeio $W^{-1} = v_kv_{k-1} \dots v_0$ é dito *passeio reverso* de W e o passeio $WW' = v_0v_1 \dots v_l$ é dito *concatenação* de W com W' . Chama-se *seção* do passeio W uma subsequência (v_i, v_j) -seção $= v_iv_{i+1} \dots v_j$ de termos consecutivos de W [5].

Se as arestas e_1, e_2, \dots, e_k de um passeio W são todas distintas — o que sempre ocorre em grafos simples — chama-se W de *trilha*. Se, adicionalmente, os vértices da trilha W forem todos distintos, chama-se W de *caminho* (também conhecido como *caminho simples* [11]).

Conektividade

Dois vértices u e v de G são ditos *conectados* se existe um (u, v) -passeio em G . A conectividade induz uma relação de equivalência sobre o conjunto de vértices V [5]: Há uma partição de V em subconjuntos não vazios $V_1, V_2, \dots, V_\omega$ tal que dois vértices u e v são conectados se, e somente se, u e v pertencem ambos ao mesmo subconjunto V_i . Os subgrafos induzidos $\langle V_1 \rangle, \langle V_2 \rangle, \dots, \langle V_\omega \rangle$ são chamados *componentes de G* . Se G tem exatamente uma única componente, então G é dito *conectado*; e, do contrário, G é dito *desconectado*.

A Figura 2.6 mostra dois grafos: O grafo da esquerda é conectado — possui uma única componente $\langle \{v_1, v_2, v_3, v_4\} \rangle$; porém, o da direita não é — pois possui duas componentes $\langle \{v_1, v_2, v_3\} \rangle, \langle \{v_4\} \rangle$.

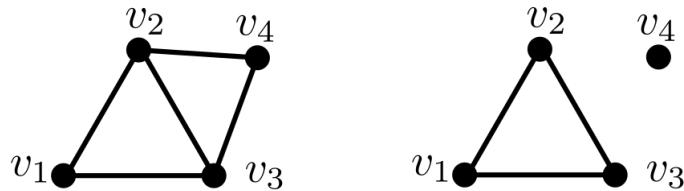


Figura 2.6: A esquerda um grafo conectado e, a direita, um grafo desconectado

Grafos Completos

Introduze-se agora uma classe especial de grafos [5]: Um grafo é dito *completo* se possui todas as suas arestas possíveis, i.e., para cada par de vértices distintos $u, v \in V_G$, u é adjacente a v (vide Figura 2.7).

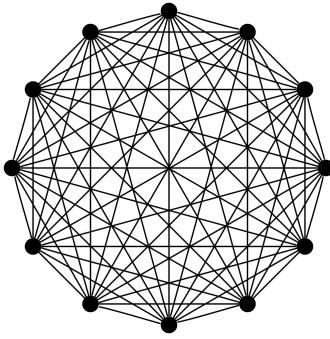


Figura 2.7: Diagrama de um grafo completo com 12 vértices ($|V| = 12$).

Usando combinatória, sabe-se que todo grafo completo com n vértices possui $\binom{n}{2} = \frac{n(n-1)}{2}$ arestas.

Em particular, chama-se de k -clique um grafo G com k vértices tal que G é completo. Por exemplo, se $k = 2$ tem-se uma linha — veja a Figura 2.8; e, se $k = 3$, forma-se um triângulo.

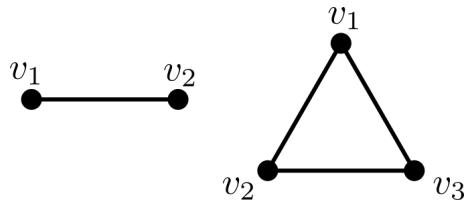


Figura 2.8: Na esquerda um 2-clique e, na direita, um 3-clique.

2.1.4 Grafos Ponderados

As arestas $e \in E$ de um grafo G pode estar associadas com um número real $d(e)$, chamado de *peso da aresta* e [5]. Quando G tem todas as suas arestas associadas com pesos, define-se G como um *grafo ponderado*. Grafos ponderados são frequentemente associados com aplicações em teoria de grafos [11].

Costuma-se definir uma *função ponderação* $d : E \rightarrow \mathbb{R}_+$ para mapear o conjunto de arestas E no conjunto dos números reais não negativos \mathbb{R}_+ [12]. Escreve-se $G = (V_G, E_G, d)$ como um grafo ponderado com o conjunto de vértices V_G , arestas E_G e função ponderação d .

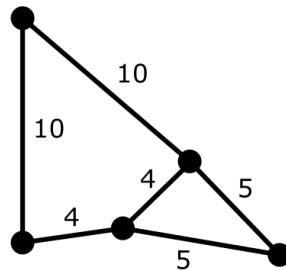


Figura 2.9: Representação de um grafo ponderado.

2.2 Geometria de Distâncias Euclidianas

Apresenta-se nesta seção uma introdução a *Geometria de Distâncias Euclidianas*. O nome “Geometria de Distâncias” diz respeito ao conceito desta geometria basear-se em distâncias ao invés de pontos. A palavra “Euclidiana” é importante para caracterizar as arestas — elementos fundamentais associados as distâncias — como segmentos, sem restringir seus ângulos de incidência [12].

2.2.1 Como tudo Começou

Por volta de 300 AC, Euclides de Alexandria organizou o conhecimento de sua época acerca da Geometria em uma obra composta por treze volumes, onde construiu, a partir de um pequeno conjunto de axiomas fortemente baseado nos conceitos de pontos e linhas, a chamada *Geometria Euclidiana* [13]. Em contraponto a visão original de Euclides, os primeiros conceitos geométricos usando *apenas distâncias* costumam estar associados aos trabalhos de Heron de Alexandria (10 a 80 DC) [12], com o desenvolvimento de um teorema que leva seu nome, como segue:

Teorema de Heron: Sejam s o *semiperímetro* de um triângulo (se p é o perímetro, $s = \frac{p}{2}$) e a, b e c os comprimentos dos três lados deste triângulo. Então, a área A do triângulo é

$$A = \sqrt{s(s-a)(s-b)(s-c)}. \quad (\text{Fórmula de Heron})$$

Pode-se dizer que esse foi o nascimento da *Geometria de Distâncias* (*Distance Geometry*, ou DG).

Algumas centenas de anos depois, em 1841, Arthur Cayley (1821 a 1895) generalizou a Fórmula de Heron através da construção de um determinante que calcula o conteúdo (volume n -dimensional) de um *simplex*¹ em qualquer dimensão [14]. Um século depois, em 1928, o matemático austríaco Karl Menger (1902 a 1985) re-organizou as ideias de Cayley e trabalhou em uma construção axiomática da geometria através de distâncias [15] — donde a alteração no nome do determinante de Cayley para como é conhecido hoje: “*Determinante de Cayley-Menger*”.

Definição: Sejam A_0, A_1, \dots, A_n $n+1$ pontos que definem os vértices de um n -simplex em um espaço euclidiano K -dimensional, onde $n \leq K$, e seja d_{ij} a distância entre os vértices A_i e A_j , onde $0 \leq i < j \leq n$. Então, o conteúdo v_n desse n -simplex é

$$v_n^2 = \frac{(-1)^{n+1}}{(n!)^2 2^n} \begin{vmatrix} 0 & d_{01}^2 & d_{02}^2 & \cdots & d_{0n}^2 & 1 \\ d_{01}^2 & 0 & d_{12}^2 & \cdots & d_{1n}^2 & 1 \\ d_{02}^2 & d_{12}^2 & 0 & \cdots & d_{2n}^2 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ d_{0n}^2 & d_{1n}^2 & d_{2n}^2 & \cdots & 0 & 1 \\ 1 & 1 & 1 & \cdots & 1 & 0 \end{vmatrix}. \quad (\text{Determinante de Cayley-Menger})$$

Mas foi só com Leonard Blumenthal (1901 a 1984) que, em 1953, o termo Geometria de Distâncias foi cunhado — com a publicação de seu livro “*Theory and*

¹Um simplex é uma generalização do conceito de triângulo a outras dimensões, i.e.: O *0-simplex* é um ponto, *1-simplex* é um segmento de reta, *2-simplex* é um triângulo e o *3-simplex* é um tetraedro.

Applications of Distance Geometry” [16]. Blumenthal dedicou sua vida de trabalho para clarificar, organizar e traduzir as obras originais em alemão [12]. Ele acreditava que o problema mais importante nesta área era o “*Problema de Subconjunto*” (ou *Subset Problem*, originalmente), que consistia em encontrar condições necessárias e suficientes a fim de decidir quando uma matriz simétrica era, de fato, uma *matriz de distâncias*² [4]. Uma restrição desse problema à métrica euclidiana chama-se *Problema de Matrizes de Distâncias Euclidianas* (ou EDMP, do inglês *Euclidean Distance Matrix Problem*), como segue definida:

Problema de Matrizes de Distâncias Euclidianas: Determinar se, para uma dada matriz quadrada $D_{n \times n} = (d_{ij})$, existe um inteiro K e um conjunto $\{p_1, \dots, p_n\}$ de pontos em \mathbb{R}^K tal que $d_{ij} = \|p_i - p_j\|$ para todo $i, j \leq n$.

Condições necessárias e suficientes para que uma matriz seja, de fato, uma matriz de distância euclidiana são dados em [17]. Para isso, apresenta-se um teorema onde se utiliza o Determinante de Cayley-Menger na criação de duas condições afirmando que, afim de $D_{n \times n}$ ser uma matriz de distâncias euclidianas, deve haver um K -simplex S de referência com conteúdo $v_K \neq 0$ em \mathbb{R}^K e que todos os $(K+1)$ -simplex e $(K+2)$ -simplex contendo S como uma das faces devem estar contidos em \mathbb{R}^K [4].

Blumenthal percebeu a importância em se respeitar as restrições métricas estabelecidas pelas matrizes de distâncias.

Quando temos como dado um conjunto de distâncias entre pares de pontos, a geometria das distâncias pode dar uma dica para encontrar um conjunto de coordenadas correto para pontos no espaço Euclídeo tridimensional, satisfazendo as restrições de distâncias dadas.

(Blumenthal, 1953, [16])

Pode-se dizer que resolver o Problema de Matrizes de Distâncias Euclidianas está intimamente relacionado com descobrir as coordenadas dos pontos que definem suas distâncias. Perceba que este é um problema inverso, onde o “problema direto” correspondente é calcular distâncias associadas a pares de pontos dados. Note que este estudo tem enorme aplicabilidade [4].

Adiante, em 1979, Yemini (atualmente professor emérito de Ciência da Computação na Universidade de Columbia) foi o primeiro a flexibilizar a definição do EDMP ao considerar um conjunto de distâncias esparso [18, 4] — i.e., que não se tem todas as distâncias dadas a priori. Com isso, introduziu-se o que se chamou de *Problema Posição - Localização*, onde deseja-se calcular a localização de todos os objetos imersos em um espaço geográfico [18].

Assim, foi possível re-formular o problema fundamental de Geometria de Distâncias, o qual pode ser caracterizado de forma mais moderna pela utilização da Teoria de Grafos [4].

²Seja o par (\mathcal{X}, d) um *espaço métrico* (vide Apêndice A), onde $\mathcal{X} = \{x_1, \dots, x_n\}$. Uma *matriz de distância* sobre \mathcal{X} é uma matriz quadrada $D_{n \times n} = (d_{uv})$ onde, para todo $u, v \leq n$, temos $d_{uv} = d(x_u, x_v)$ [4].

2.2.2 O Problema Fundamental

Uma *realização* é uma função que mapeia um conjunto de vértices de um grafo G para um espaço euclidiano de alguma dimensão dada [12].

Problema de Geometria de Distâncias (DGP): Dados um grafo simples, ponderado e conectado $G = (V, E, d)$ e um inteiro $K > 0$, encontre uma realização $x : V \rightarrow \mathbb{R}^K$ tal que:

$$\forall \{u, v\} \in E, \quad \|x(u) - x(v)\| = d(u, v). \quad (2.1)$$

Desde que uma realização seja encontrada, também dá-se a ela o nome de *solução* do DGP. Por simplicidade — claramente um abuso de notação —, pode-se escrever x_u e d_{uv} no lugar de $x(u)$ e $d(u, v)$, respectivamente.

A principal diferença desta definição para o EDMP está acerca de que uma matriz de distância essencialmente representa um *grafo ponderado completo*. Em contraponto, o DGP não empoe qualquer estrutura em G^3 , seguindo o conceito de matriz esparsa estabelecido por Yemini.

Por fim, na equação 2.1, utiliza-se a norma euclidiana $\|\cdot\|$ como métrica (ver Apêndice A), donde pode-se reescrever esta equação como

$$\forall \{u, v\} \in E, \quad \sqrt{\sum_{i=1}^K (x_{ui} - x_{vi})^2} = d_{uv}.$$

Como a definição de métrica garante a positividade das distâncias, pode-se esconder a raiz quadrada na equação acima, i.e.

$$\forall \{u, v\} \in E, \quad \sum_{i=1}^K (x_{ui} - x_{vi})^2 = d_{uv}^2. \quad (2.2)$$

2.2.3 Os Diferentes Problemas em DG

Em 2014, Leo Liberti *et al.* publicaram um ótimo compendio sobre a *Geometria de Distâncias Euclidianas e suas Aplicações* [4] e, em particular, desenvolveram um estudo taxonômico muito interessante sobre os problemas clássicos da área. No que se segue, devido a grande quantidade de siglas e variações dentro de DG, apresenta-se parte desse estudo, visando organizar os conceitos.

As principais aplicações em DG são no *calculo de estruturas moleculares* [19], na *localização de sensores em rede sem fio* (*Wireless Sensor Network Localization*, ou WSNL) [20], em *cinematática inversa* (*Inverse Kinematic*, ou IK) [21] e em *escalonamento multidimensional* (*Multidimensional Scaling*, ou MDS) [22].

Conformações Moleculares

Existe uma relação muito forte com a forma geométrica das moléculas e suas funções em organismos vivos [23]. Projetar drogas para curar uma doença específica

³A menos, é claro, no que diz respeito a seus vértices estarem conectados. Porém, caso G não seja conectado, então ele consiste de um conjunto de diferentes subgrafos conectados, donde, a fim de solucionar o DGP, pode-se realizar cada subgrafo separadamente [12].

se trata basicamente de conhecer o que uma certa proteína pode fazer em um organismo [12]. Proteínas se ligam em outras moléculas através do equilíbrio de forças agindo entre elas⁴, por tanto, suas ligações dependem do seu formato.

Proteínas são constituídas por um grande conjunto de átomos e, alguns pares destes, trocam ligações químicas — sabe-se quais são esses átomos através de experimentos de cristalografia [25]. Então, se os átomos de uma molécula forem rotulados da forma $1, 3, 4, \dots, n$, então é possível inferir:

- O conjunto de ligações $\{u, v\}$, onde u, v são átomos em $\{1, \dots, n\}$;
- A distância entre u e v (para cara par ligado);
- O ângulo interno θ_v definido por duas ligações $\{u, v\}$ e $\{v, w\}$, com um átomo v em comum.

Além desses dados, também é possível obter informações a partir de experimentos mais sofisticados, como a *Ressonância Magnética Nuclear* (RMN). Neste experimento é escolhida uma faixa de radiofrequência para bombardear uma amostra que está imersa em um campo magnético bastante intenso. Dependendo da radiofrequência utilizada (costuma-se usar a do hidrogênio), alguns núcleos atômicos irão absorver energia e outros não. Caso atinja-se uma frequência exata de ressonância dentro destes núcleos atômicos, é possível medir essa ressonância como um sinal de radiofrequência enviado dos núcleos atômicos — para calcular distâncias entre átomos próximos, com distâncias menores que 5 Å.

De posse dessas informações, deseja-se realizar (localizar) todos os átomos da molécula. Esse problema, com todas as informações moleculares disponíveis, denomina-se *Estrutura Proteica a partir de Dados Brutos* (*Protein Structure from Raw Data*, ou PSRD)

Em particular, como as coordenadas atômicas pertencem ao \mathbb{R}^3 , há uma particularização do DGP para o caso molecular, chamado *Problema de Geometria de Distâncias Moleculares* (*Molecular DGP*, ou MDGP). Trata-se do DGP com $K = 3$ fixo.

Localização de Sensores

O *Problema de Localização de Sensores em Rede sem Fio* (ou *WSNL Problem*) surge quando é necessário localizar um conjunto de objetos equipados com sensores eletrônicos capazes de medir distâncias entre si, geograficamente distribuídos, usando apenas medidas de distâncias entre pares destes objetos [20].

Por exemplo, *smartphones* com WIFI ativo podem criar uma rede conhecida por *Rede Ad-Hoc*, i.e., eles conseguem criar uma rede para comunicar-se entre si, de forma *peer-to-peer*, sem a necessidade de uma torre central — cada aparelho funciona como uma pequena torre, de forma que a distância entre os aparelhos não pode ser excessiva. Dessa forma, os *smartphones* podem estimar a distância r de emparelhamento das suas conexões ao medir, por exemplo, qual a potência de

⁴Ou seja, o equilíbrio da energia potencial das moléculas, proporcional, principalmente, as variações nos comprimentos das ligações covalentes, as variações nos ângulos entre duas ligações covalentes consecutivas, as rotações sobre as ligações covalentes e as interações de van der Waals e interações eletrostáticas entre átomos [24].

transmissão do sinal, uma vez que sabe-se que a potência P de uma transmissão eletromagnética cai da forma

$$P = \frac{X}{r^n}, \quad (2.3)$$

onde X e n são constantes e dependem muito das condições do experimento, sendo obtidas experimentalmente [26].

Em essência, um problema do tipo WSNL segue a mesma definição do DGP, porém, com um subconjunto $A \subset V$ de vértices (chamados *âncoras*), onde os elementos de A tem uma posição em \mathbb{R}^k dada a priori — isso é feito pois, normalmente, interessa saber a posição relativa de um objeto a outro, como é o caso do Sistema de Posicionamento Global, onde temos os satélites como âncoras e desejamos saber a posição dos aparelhos GPS em relação aos satélites.

Por motivos práticos — semelhantes ao caso molecular — as variações de interesse desse problema tem o K fixo em $K = 2$ ou $K = 3$. É comum, também, que se defina um WSNL como *solucionável* somente se seu grafo possua uma única realização válida [12] — noção conhecida como *globalmente rígido*: Diz-se que um grafo é *globalmente rígido* quando ele possui uma realização genérica x e, para todas as outras realizações x' , x é congruente a x' .

Dinâmicas em Cinemática Inversa

Muito utilizada em robótica e animação computadorizada, a cinemática inversa cerne sobre mecanismos e seus movimentos rígidos, onde restringe-se os movimentos de forma a preservar a geometria do sistema. Sem o auxílio computacional e matemático a manipulação de mecanismos com muitos graus de liberdade pode ser inviável: Imagine a manipulação manual de cem vértices em uma haste simulando o comportamento de um braço articulado em uma animação. Com o auxílio da DG, um animador pode apenas configurar a posição final de um pequeno grupo de vértices (como os da extremidade da aresta, por exemplo) e um algorítimo de cinemática inversa é capaz de verificar se aquela posição é ou não viável e, se viável, qual a realização de todo o conjunto de vértices em razão da posição configurada [21].

Visando tal restrição mecânica, define-se o *Problema de Cinemática Inversa (Inverse Kinematic Problem, ou IKP)* como uma variação do WSNL — logo, tem o objetivo de descobrir posições em relação a certos pontos previamente realizados — com uma restrição no grafo que define o problema: deve ser um caminho simples com seus vértices finais sempre sendo âncoras [4].

Escalonamento Multidimensional

O problema de *Escalonamento Multidimensional (Multidimensional Scaling, ou MDS)* é definido como [4]: Dado um conjunto X de vetores, encontre um conjunto Y de vetores com menor dimensão (com $|X| = |Y|$) tal que a distância entre cada i -ésimo e j -ésimo vetores de Y tenham, aproximadamente, a mesma distância que seus pares de vetores correspondentes em X .

Esse problema é muito aplicado na análise de dados em Big Data [12]. É um meio de facilitar a visualização do nível de similaridade entre casos individuais — que não necessariamente precisam ter uma conexão aparente — em um conjunto de dados. Pode-se usá-lo, por exemplo, para visualizar em uma escala bidimensional

(\mathbb{R}^2) a evolução da locomoção de animais no espaço tridimensional utilizando dados de séries temporais (espaço em diferentes tempos, logo, dados em \mathbb{R}^4).

2.2.4 A Busca de uma Solução

A abordagem mais simples, pode-se pensar, para encontrar um conjunto de soluções que satisfação a equação 2.2 é resolver o sistema de equações diretamente [27]. Infelizmente, para $K \geq 2$, há evidências de que uma solução de forma fechada onde todo componente de x é expresso por raízes, não é possível [12].

No entanto, pode-se re-formular o problema como um problema de otimização global [12], onde o objetivo é minimizar a soma dos *erros*⁵ entre as distâncias dadas a priori e as calculadas. Para isso, pode-se considerar uma única expressão que englobe todos os n erros, da forma

$$f(x_1, \dots, x_n) = \sum_{(i,j) \in E} (\|x_i - x_j\| - d_{ij})^2. \quad (2.4)$$

Fica claro que encontrar uma solução para o DGP é equivalente a encontrar realizações $x_i \in \mathbb{R}^3$, $i = 1, \dots, n$, se e somente se $f(x_1, \dots, x_n) = 0$ [12]. Pela definição de métrica (vide Apêndice A), 0 é o menor valor possível para f , donde diz-se que deseja-se *minimizar a função* $f : \mathbb{R}^n \rightarrow \mathbb{R}$ [27]. Ou seja,

$$\min_{x_i \in \mathbb{R}^n} f(x_1, \dots, x_n). \quad (2.5)$$

E, no caso da métrica euclidiana (vide Apêndice A), temos

$$\min_{x_j \in \mathbb{R}^n} \sum_{(u,v) \in E} \left(\sum_{i=1}^K (x_{ui} - x_{vi})^2 - d_{uv}^2 \right)^2. \quad (2.6)$$

Por tanto, a equação 2.6 tem como objetivo a minimização de um polinômio de múltiplas variáveis de grau quatro.

Um dos desafios da Otimização Global é que muitos dos métodos existentes — em especial, os mais eficientes — não garantem que uma otimização *global* será encontrada [12]. Isso se dá pois, dependendo do comportamento da função, existem muitos ótimos locais e os métodos não conseguem diferenciá-los de um global.

Infelizmente, essa abordagem via otimização é custosa do ponto de vista computacional [12]. Saxe demonstrou em 1979 [28] que resolver um DGP para qualquer dimensão — i.e., para qualquer valor de K — tem a complexidade computacional da classe **NP-Hard**. Em outras palavras, isso significa que a quantidade de mínimos locais de um DGP cresce exponencialmente proporcional a $|V|$ [29].

A Quantidade de Soluções do Problema

Dada uma solução de um DGP, obtém-se facilmente uma quantidade infinita (não enumerável!) de outras realizações válidas, distintas, através de rotações e translações desta solução inicial [27], devido ao fato destes movimentos serem rígidos, donde não deformam as distâncias associadas aos pares de vértices. Por descreverem uma

⁵Em otimização, vê-se a equação 2.1 de forma não exata: $\|x_u - x_v\| = d_{uv} + \varepsilon$, onde ε é chamado *erro*. Ou seja, para minimizar o erro, precisa-se minimizar a expressão $f(x_u, x_v) = \|x_u - x_v\| - d_{uv}$.

mesma geometria, essa realizações não são interessantes. Contudo, se for desconsiderado essas redundâncias, a quantidade de soluções do DGP depende da estrutura geométrica do grafo que a define: (i) podem não haver nenhuma realização; (ii) uma única realização; (iii) uma quantidade finita (não única) de realizações; ou, (iv) um número incontável de realizações [12]. Perceba que, curiosamente, a quantidade de soluções de um DGP somente não pode ser um número infinito e enumerável — sabe-se isso através de estudos em *Geometria Algébrica Real* [30].

Ou seja, supondo que o conjunto solução de um DGP seja não vazio, sabe-se que ele é não enumerável ou finito. Se for finito (normalmente o caso desejado), além de aplicar os métodos de Otimização Global — já definidos como custosos computacionalmente —, pode-se explorar outras abordagens, como a Otimização Combinatória [27].

2.2.5 Combinatória do DGP

Nesta seção, estuda-se sobre as condições que garantem a finitude do conjunto solução do problema ao analisar o espaço de busca por uma solução. Em particular, para um DGP definido em um espaço de dimensão K , apresenta-se uma classe de grafos com propriedades muito interessantes: dos $(K+2)$ -cliques, ou seja, dos grafos completos com dois vértices a mais do que o número de dimensões do seu espaço.

Realização de Grafos Completos

Dado um $(K+1)$ -clique com vértices não coincidentes (essa condição ficará mais clara ao decorrer desta seção), sabe-se que *se ele possui* uma realização em \mathbb{R}^{K-1} , *ela é única* a menos de rotações e translações [12, 31, 32]. Porém, visto que para solucionar um DGP interessa uma solução em \mathbb{R}^K ao invés de \mathbb{R}^{K-1} , pode-se usar um corolário um tanto óbvio da afirmação anterior: um $(K+2)$ -clique com vértices não coincidentes tem, no máximo, uma realização única no espaço \mathbb{R}^K (novamente, *única* a menos de movimentos rígidos).

Por tanto, dependendo da estrutura do grafo que define um DGP (em especial, dependendo de sua rigidez [2, 33]), obter uma solução do problema pode garantir a unicidade desta solução. Essa característica é de fundamental importância para algumas classes de problemas em DG, como o caso da WSNL, onde somente realizações únicas são consideradas como soluções. Ainda assim, é preciso apresentar um método para obter uma solução. É o que se faz a seguir, baseado em [12] e [34].

Considere um 3-clique ponderado com $V = \{1, 2, 3\}$, onde $d_{12} = d_{23} = 1$ e $d_{13} = 2$. Então, uma possível realização sobre a reta real \mathbb{R} que satisfaça todas as distâncias é $x_1 = 0$, $x_2 = 1$ e $x_3 = 2$. Uma forma de obter o valor de x_3 , dado os valores de x_1 e x_2 e as distâncias d_{13} e d_{23} , é a *trilateração*.

Trilateração

Sabendo que $d_{13} = \|x_3 - x_1\| = 2$ e $d_{23} = \|x_3 - x_2\| = 1$ e que, usando a norma euclidiana⁶, $\|u - v\|^2 = (u - v)^2 = u^2 - 2uv + v^2$, tem-se

⁶Ver Apêndice A

$$x_3^2 - 2x_1x_3 + x_1^2 = 4 \quad \text{e} \quad (2.7)$$

$$x_3^2 - 2x_2x_3 + x_2^2 = 1. \quad (2.8)$$

Subtraindo a equação 2.8 da 2.7, obtém-se

$$2(x_1 - x_2)x_3 = x_1^2 - x_2^2 - 3 \Rightarrow 2x_3 = 4 \Rightarrow x_3 = 2.$$

Pode-se generalizar esse exemplo facilmente para $(K+2)$ -cliques em \mathbb{R}^K [12]: Precisa-se conhecer a posição de $K+1$ vértices e as distâncias destes ao $(K+2)$ -ésimo vértice, assim, pode-se realizar o $(K+2)$ -ésimo vértice, em tempo linear, resolvendo um sistema de $K+1$ equações como acima.

Isto é, sejam $x_1, \dots, x_{K+1} \in \mathbb{R}^K$ as posições para $K+1$ vértices de um $(K+2)$ -clique e, para todo $j \leq K+1$, seja $d_{j,K+2}$ a distância associada com a aresta $\{j, K+2\}$. Seja $y \in \mathbb{R}^K$ a posição do $(K+2)$ -ésimo vértice; então, y deve respeitar as $K+1$ equações quadráticas $\forall j \leq K+1$, $\|x_j - y\|^2 = d_{j,K+2}^2$, com K incógnitas y_1, \dots, y_K :

$$\begin{cases} \|y\|^2 - 2x_1y + \|x_1\|^2 = d_{1,K+2}^2 \\ \vdots \\ \|y\|^2 - 2x_{K+1}y + \|x_{K+1}\|^2 = d_{K+1,K+2}^2 \end{cases} \quad (2.9)$$

Para qualquer $h \leq K+1$, seja e_h a h -ésima equação no sistema de equações 2.9: tomando todo $h < K+1$, pode-se obter as K diferenças $(e_h - e_{K+1})$ e formar um novo sistema, contendo K equações com K incógnitas:

$$\begin{cases} 2(x_1 - x_{K+1}) \cdot y = \|x_1\|^2 - \|x_{K+1}\|^2 - d_{1,K+2}^2 + d_{K+1,K+2}^2 \\ \vdots \\ 2(x_K - x_{K+1}) \cdot y = \|x_K\|^2 - \|x_{K+1}\|^2 - d_{K,K+2}^2 + d_{K+1,K+2}^2 \end{cases} \quad (2.10)$$

Note que o sistema de equações 2.10 é um *sistema linear* da forma

$$Ay = b, \quad (2.11)$$

onde $A = (2a_{ij})$ é uma matriz quadrada $K \times K$ com $a_{ij} = x_{ij} - x_{K+1j}$ para todo $i, j \leq K$, e $b = (b_1, \dots, b_K)^T$ com $b_i = \|x_i\|^2 - \|x_{K+1}\|^2 - d_{i,K+2}^2 + d_{K+1,K+2}^2$ para todo $i \leq K$.

Diferentes métodos para solução de sistemas lineares como a equação 2.11 são encontrados na bibliografia [35] — no geral, a escolha do melhor depende de propriedades da matriz A , como sobre sua esparsidade. Em especial, se A não é uma matriz singular, então ela possui uma inversa A^{-1} . Pode-se, por tanto, obter a posição do $(K+2)$ -ésimo vértice fazendo

$$Ay = b \Rightarrow A^{-1}Ay = A^{-1}b \Rightarrow y = A^{-1}b = x_{K+2}. \quad (2.12)$$

No entanto, se A é singular, isso quer dizer que as linhas $a_i = x_i - x_{K+1}$ (para $i \leq K$) não são todas linearmente independentes [35]. Essa situação mostra algumas propriedades geométricas interessantes [12]. Por exemplo, se $K = 1$, significa que $x_1 - x_2 = 0 \Rightarrow x_1 = x_2$, ou seja, que o segmento entre x_1 e x_2 é um simples ponto. Como estamos imersos no $\mathbb{R}^K = \mathbb{R}$ (i.e., a reta real), geometricamente, a situação é que ou x_3 está posicionado a direita ou a esquerda de $x_1 = x_2$, mas não se pode escolher. Não obstante, se $K = 2$, a singularidade de A implica que o triângulo definido por x_1, x_2 e x_3 é apenas um segmento no plano (caso o rank de A é 1) ou um simples ponto (caso o rank for 0). No primeiro caso, x_4 pode estar posicionado em ambos os lados da linha que contém o segmento e, no segundo caso, x_4 pode estar em qualquer um dos pontos formados pela circunferência com centro $x_1 = x_2 = x_3$ e raio $d_{14} = d_{24} = d_{34}$. Essa característica geométrica vale para valores maiores de K : a singularidade de A está relacionada a existência de vértices coincidentes e implica que há sempre múltiplas soluções para x_{K+2} .

Por fim, é importante mencionar que a partir da equação 2.9 podemos chegar no sistema linear 2.11, mas a volta não é verdadeira. Em particular, se o sistema 2.9 tem uma solução, então o sistema 2.11 tem a mesma solução. Porém, mesmo que o sistema 2.9 não tenha solução, o sistema 2.11 sempre terá uma solução única — desde que A não seja singular [12]. Sendo assim, para verificar a factibilidade de uma solução x_{K+2} advinda do sistema linear 2.11, deve-se verificar se as distâncias aos $K+1$ vértices foram respeitadas — ou seja, se $\|x_i - x_{K+2}\| = d_{i,K+2}$, para todo $i \leq K+1$.

Devagar e Sempre

Utilizando o método da trilateração apresentado, é possível descobrir a posição de apenas um vértice de um grafo completo. No entanto, como o objetivo é uma realização total do grafo, a seguir relembrar-se uma característica dos grafos completos que contorna essa limitação de uma forma engenhosa.

Seja o primeiro grafo completo da Figura 2.10 formado pelo conjunto de vértices $\{v_1, v_2, v_3, v_4\}$. Perceba que esse é um 4-clique e, ao removermos o vértice v_4 , obtemos um 3-clique formado pelo conjunto de vértices restantes $\{v_1, v_2, v_3\}$ (no centro da Figura 2.10). Caso for retirado o vértice v_3 desse 3-clique, obtemos o 2-clique formado por $\{v_1, v_2\}$ (ultimo grafo da Figura 2.10).

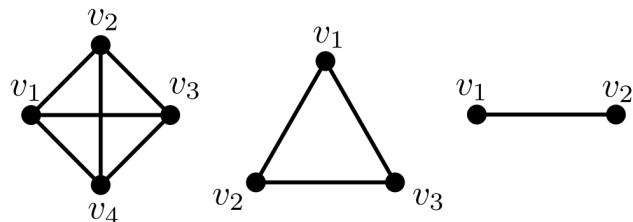


Figura 2.10: Em ordem: 4-clique; 3-clique; 2-clique.

Perceba a existência de uma estrutura recursiva nos grafos completos, que se mantém para o caso geral: Sejam $G_{K+1} = \{V_G, E_G\}$ um $(K+1)$ -clique e $v \in V_G$

um vértice qualquer de G_{K+1} , sempre pode-se obter um K -clique como o subgrafo induzido $\langle V_G \setminus \{v\} \rangle$. Isto é, se (V, E) é um grafo completo, dado qualquer grafo induzido $\langle V' \rangle$ com $V' \subset V$, o subgrafo $\langle V' \rangle$ também é completo. Por conta disso, podemos utilizar essas estruturas como “blocos básicos de construção” para planejar uma realização iterativa do grafo como um todo, usando a trilateração para realizar um novo vértice em cada iteração.

Realização Iterativa de Grafos Completos

A seguir, apresenta-se um algorítimo para realizar em \mathbb{R}^K todos os n vértices de um $(n, \frac{n^2-n}{2})$ -grafo completo G , com $n > K$, tendo como entrada a posição dos $(K+1)$ primeiros vértices também em \mathbb{R}^K .

Primeiro, assume-se que existe um $(K+1)$ -clique $G_o \subset G$, chamado clique inicial, que conhecemos a realização — em WSNL, por exemplo, comumente se utiliza nós ancoras como clique inicial [2, 26]. Sem perda de generalidade, seja $\{1, \dots, K+1\}$ o conjunto dos vértices que formam a clique inicial G_o , com realizações $\{x_1, \dots, x_{K+1}\}$. Seja, também, $N(i)$ o conjunto de vértices adjacentes ao i -ésimo vértice. Então, pode-se encontrar uma realização total de G através do Algorítimo 1.

Algorithm 1: $x = \text{RealizacaoIterativa}(G, d, K, x)$ [12]

```

// Realize os próximos vértices iterativamente
1 for  $i \in \{K+2, \dots, n\}$  do
    /* Utilize o  $(K+1)$ -clique dos  $(K+1)$  antecessores imediatos
       de  $i$  para calcular a realização  $x_i$ . Caso não haja
       solução, atribuir  $\emptyset$  */ 
    2  $x_i = \text{Trilateracao}(x_{i-K-1}, \dots, x_{i-1})$ ;
       // verifique se  $x_i$  é factível com relação as demais
       distâncias
    3 for  $\{j \in N(i) ; j < i\}$  do
        4 if  $\|x_i - x_j\| \neq d_{ij}$  then
            // Sinalizar como não factível e sair do loop
            5  $x_i = \emptyset$ ;
            6 break;
        7 end
    8 end
    9 if  $x_i = \emptyset$  then
        // Retornar que a realização não é factível
    10 return  $\emptyset$ ;
    11 end
12 end
// Retornar a realização factível
13 return  $x$ ;

```

Note que o Algorítimo 1 tem a complexidade de seu pior caso como $O(K^3n)$, i.e., para todos os n vértices, deve-se resolver um sistema linear $K \times K$ (trilateração). Se não existir realização factível para G em \mathbb{R}^K , Algorítimo 1 retorna \emptyset .

Esse processo de trilateração em \mathbb{R}^K é chamado K -*lateração* [2, 12].

Sobre o clique inicial G_o e unicidade

O Sistema de Posicionamento Global (GPS, [36]) é um exemplo de WSNL que utiliza da trilateração para descobrir a localização dos aparelhos de GPS (sensores móveis). Como o objetivo é encontrar posições no \mathbb{R}^3 , precisa-se de 4 vértices âncoras para compor o clique inicial G_o , que, no caso, é formado por um conjunto de satélites com posições bem conhecidas. Fica claro que, em algumas aplicações, a quantidade de vértices necessários no clique inicial pode significar um projeto de engenharia bastante custoso.

Além disso, em um primeiro momento pode parecer pouco razoável necessitar da realização prévia do clique inicial G_o . De fato, se o problema possuir apenas $K + 1$ vértices, essa solução não faz sentido. Felizmente, em geral, os problemas de estudo costumam ser maiores [4].

Bem, como dito anteriormente, na trilateração, a necessidade de se usar um vértice a mais do que a dimensão do espaço de realização está relacionada com a unicidade da solução [31, 32]. Perceba, os $K + 1$ vértices do clique inicial mais o vértice a se realizar induzem um simplex no espaço \mathbb{R}^K que, garantida a desigualdade triangular, possui um volume K -dimensional ≥ 0 diretamente proporcional ao determinante de Cayley-Menger. Caso esse volume for zero, que é o que acontece com $(K+2)$ -simplex em \mathbb{R}^K , tem-se o chamado *simplex achatado* (flat simplex) com no máximo uma solução [12] (como ilustra a Figura 2.11, a direita).

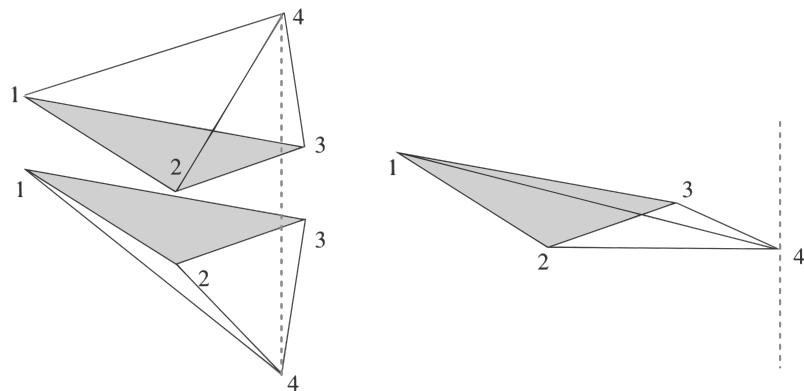


Figura 2.11: Note que as duas realizações de um mesmo 4-clique em \mathbb{R}^3 (esquerda) são possíveis por respeitarem as distâncias entre os vértices, mas somente uma realização é possível em \mathbb{R}^2 (direita) [12].

Em particular, a noção que estuda a unicidade de uma realização é a de rigidez: diz-se que um grafo é *globalmente rígido* se ele tem uma realização genérica x e, para todas as outras realizações x' , x é *congruente* a x' (veja Apêndice B). Um grafo globalmente rígido tem solução única [4, 37].

Realizando grafos K -laterativos em \mathbb{R}^K

No Algorítimo 1, fica implícita a existência de uma ordem no conjunto de vértices V do grafo G . Se G é completo, de fato, qualquer ordem (v_1, \dots, v_n) em V é tal que v_i é adjacente a todos os seus predecessores — isto é, para todo $i > K + 1$, tem-se

no mínimo os $K + 1$ predecessores necessários para a trilateração. Por outro lado, G não precisa ser necessariamente completo para garantir isso [12].

Definição: Se $<$ é uma ordem sobre V e $v \in V$ é um vértice qualquer, então $\rho(v)$ é a posição de v em relação a $<$ e $\gamma(v)$ é o conjunto de predecessores de v em relação a $<$. Dado um grafo $G = (V, E)$, uma ordenação $<$ sobre V é chamada *ordem de K-lateração* se:

1. os primeiros $K + 1$ vértices de $<$ induzirem um $(K + 1)$ -clique G_o em G ;
2. todo vértice v , com $\rho(v) > K + 1$, tem $|N_G(v) \cap \gamma(v)| \geq K + 1$.

Um grafo $G = (V, E)$ é dito *K-laterativo* se há uma ordem de K -lateração sobre V associada a G . Perceba que um grafo K -laterativo não precisa ser completo e, mesmo assim, ainda é possível aplicar a trilateração em todo vértice $v \in V$, com $\rho(v) > K + 1$ seguindo a ordem de K -lateração, pois esta garante que sempre existirá um $(K + 1)$ -clique realizado adjacente a v . O que nos leva ao enunciado principal dessa seção:

Teorema: Um grafo K -laterativo em \mathbb{R}^K é genericamente globalmente rígido em \mathbb{R}^K [2, 4]. Ou seja, se possuir realização, ela é única.

A partir desses conceitos, no que se segue define-se uma subclasse do problema central e uma adaptação do Algorítimo 1 para solucioná-lo.

O Trilaterativo DGP (TDGP): Um DGP (G, d, K) é chamado *Trilaterativo DGP* se uma ordem de K -lateração sobre G for dada.

Dado um TDGP (G, d, K) , seja $\{x_1, \dots, x_{K+1}\}$ o conjunto de realizações dos primeiros $K + 1$ vértices em relação a ordem de K -lateração. Uma realização x de G em \mathbb{R}^K pode ser encontrada (ou mostrada que não existe) pelo Algorítimo 2.

Algorithm 2: $x = \text{RealizacaoTrilaterativa}(G, d, K, x)$, adaptado de [12]

```

1 for  $i \in \{K + 2, \dots, n\}$  do
2   // Procure os primeiros  $K + 1$  predecessores adjacentes
3   sejam  $U \subset |N_G(v) \cap \gamma(v)|$ , com  $|U| = K + 1$ , e  $W = \{x_j \mid j \in U\}$ 
4   // Utilize o  $(K + 1)$ -clique definido por  $W$  para realizar  $x_i$ 
5    $x_i = \text{Trilateracao}(W);$ 
6   for  $\{j \in \{(N_G(v) \cap \gamma(v)) \setminus U\} ; j < i\}$  do
7     if  $\|x_i - x_j\| \neq d_{ij}$  then
8        $x_i = \emptyset;$ 
9       break;
10      end
11    end
12    if  $x_i = \emptyset$  then
13      return  $\emptyset;$ 
14    end
15  end
16  return  $x;$ 

```

Perceba que há três características que fazem desta uma boa solução para instâncias WSNL: (i) O $(K + 1)$ -clique inicial necessita de uma realização dada a priori; (ii) sempre possuirá ou nenhuma (se não existe realização em \mathbb{R}^K), ou exatamente uma solução; (iii) é resolvido em tempo polinomial [12].

Capítulo 3

Resultados e Discussão

3.1 Robótica Móvel

Um *robô móvel*, como o nome sugere, tem a habilidade de se locomover. São contrastantes com os chamados *robôs industriais*, ou *robôs de manipulação com base fixa*, comumente aplicados em grandes indústrias como soluções de automação — potentes braços mecânicos empregando movimentos repetitivos, como a montagem de automóveis, aviões, montagem de componentes eletrônicos, pintura e outros. São diversos os tipos de robôs móveis: podem viajar em ambientes fechados, no solo, na superfície de fluidos, por baixo de fluidos, no ar e até no espaço. Essa capacidade de mobilidade fez destes soluções para vários problemas complicados de engenharia, elevando consideravelmente a pesquisa da área nas últimas décadas [1].

Existem várias classificações diferentes para robôs móveis [3] (veja a Figura 3.1). Robôs terrestres são distinguidos principalmente por *Robôs Móveis com Rodas* (WMRs, do inglês *Wheeled Mobile Robots*) e *Robôs Móveis com Pernas* (LMRs, de *Legged Mobile Robots*). Também existem os *Veículos Aéreos Não Tripulados* (VANTs, ou UAVs de *Unmanned Aerial Vehicles*, popularmente conhecidos como *drones*) e os *Veículos Subaquáticos Autônomos* (AUVs, de *Autonomous Underwater Vehicles*).



Figura 3.1: Exemplos de robôs móveis. Em ordem: Um WMR usado para navegar pelo terreno de marte; Um LMR que imita o comportamento canino; Um VANT, drone com quatro hélices; Um AUV usado em elevadas profundezas marítimas.

Os WMRs são muito populares devido sua baixa complexidade mecânica e pouco consumo de energia [38], porém, os VANTs ganharam parte interessante do mercado devido o barateamento pela fabricação em escala industrial e grande interesse da po-

pulação. Há empresas, por exemplo, aplicando a utilização de VANTs como soluções de logística para entregas [39]. Uma interessante apresentação histórica da evolução dos robôs móveis é feita em [3].

De qualquer forma, como nesse texto tem-se um interesse geral por robôs móveis, não necessariamente limitados a uma aplicação específica, recorrentemente os abstrairemos como *sensores*. Veja, como geralmente robôs móveis devem executar suas tarefas de forma autônoma, é de extrema importância para sua supervisão, funcionamento e controle o conhecimento de suas posições [1]. Uma vez que estabelece-se uma coordenada de destino, é de extrema importância que estes utilizem de seus conjuntos de sensores para se localizar e estabilizar sua navegação.

3.1.1 Sensores

Variados tipos de sensores são comumente empregados para auxiliar na locomoção de robôs [40]. Existem sensores de controle local, que auxiliam na locomoção do robô, garantindo que este se mantenha no melhor trajeto e não cause danos a si mesmo ou ao ambiente ao seu redor; e existem os sensores de controle global, que orientam o robô no espaço.

- **Controle Local**

À distância: Utilizados para determinar a presença (e estimar distância) de obstáculos e objetos sólidos no geral. Pode-se mencionar sensor infravermelho, de ultrassom, radar, radar laser e sistemas de visão computacional como exemplos;

Por contato ou proximidade: Informam a presença de um objeto ao alcance do robô. Alguns exemplos são bumpers, switches, animal whiskers, sensores eletromagnéticos, indutivos e capacitivos.

De deslocamento e velocidade: Possibilitam a predição da posição e orientação *relativa* do robô. Sensores de inércia (giroscópio e acelerômetros), odômetros (encoders ópticos ou por contato; contadores de giro), sensores de efeito doppler, trilateração não ancorada e sensores baseados em visão computacional também são comuns.

- **Controle Global**

De posição e orientação: Determinam a posição e orientação *absoluta* do robô. Módulos de GPS (Sistema de Posicionamento Global, trilateração por satélites), bússolas, módulos de trilateração usando âncoras, visão computacional (indoor) e guias fixas pré-estabelecidas (como faixas pintadas no chão) são técnicas muito empregadas.

Nota-se que visão computacional pode ser aplicada para resolver vários dos problemas de engenharia envolvendo navegação autônoma. De fato, é com essa técnica que muitos dos animais se baseia para a locomoção. Infelizmente, esta ainda é uma solução muito custosa computacionalmente, se tornando inviável para a muitas aplicações embarcadas [41] e que necessitam de um controle em tempo real (críticas).

Em específico, nesse texto, interessa aplicar conceitos de Geometria de Distâncias no controle de *posicionamento absoluto* de um robô móvel. Para isso, baseando em

[2] e [26], define-se o problema como um grafo ponderado $G = (V, E, d)$, onde V é formado pelo conjunto de sensores no espaço e E define o conjunto de sensores que estão hábeis a se comunicar, respeitando suas limitações, e as distâncias entre si.

3.1.2 Como medir distâncias

Os métodos mais populares para estimar distância entre dois sensores são [26]: *Indicador de força de sinal recebido* (RSSI, de *Received Signal Strength Indicator*) e *Métodos baseados em tempo*.

Intensidade de Sinal

Seja por fio ou não, a comunicação entre sensores móveis é feita por ondas eletromagnéticas que tem sua potência alterada entre o envio e recebimento. Isso se dá pois a intensidade de um sinal eletromagnético I é definido como a potência da fonte P sobre área de propagação A , i.e., $I = \frac{P}{A}$ [42]. Supondo que se utilize comunicação sem fio, então o sinal é propagado pelo ar em todas as direções. Como estamos no \mathbb{R}^3 , a área de propagação do sinal é a superfície da esfera com centro na fonte, ou seja,

$$A(r) = 2 \int_0^{\frac{\pi}{2}} 2\pi r^2 \cos(\theta) d\theta = 4\pi r^2 \Rightarrow I = \frac{P}{4\pi r^2}$$

Por tanto, a intensidade de sinal recebido é inversamente proporcional ao quadrado da distância entre a fonte do sinal e o receptor. Fixando a potência do transmissor e medindo a intensidade do sinal recebido, pode-se estimar a distância r . Esse é o processo baseado em RSSI.

Savvides *et al.* [26] desenvolveram um protótipo para testar a eficiência desse método e, infelizmente, não obtiveram bons resultados. Constataram que existem muitas fontes de interferência na intensidade do sinal eletromagnético, implicando em muitas condições sobre o funcionamento. Por exemplo, a estimativa muda se houver obstáculos entre os sensores, se estiverem em um ambiente fechado ou aberto, se houver vegetação e se houver diferença de altura.

Atraso no sinal

Outra característica da comunicação entre dois sensores é que há um tempo para que o sinal saia do transmissor e chegue no receptor. Visto que os sinais eletromagnéticos são transmitidos na velocidade da luz, de fato, esse atraso da comunicação não é facilmente mensurável para curtas distâncias. Porém, pode-se utilizar outras fontes de sinal com velocidades menores afim de medir esse atraso. Esse é o princípio do sensor ultrassônico (vide Figura 3.2), que emite uma onda sonora em alta frequência (inaudível por nós, mas com velocidade constante de $\approx 340\text{m/s}$) e mede o tempo que a onda demora para ecoar em objetos e voltar para o sensor [40].

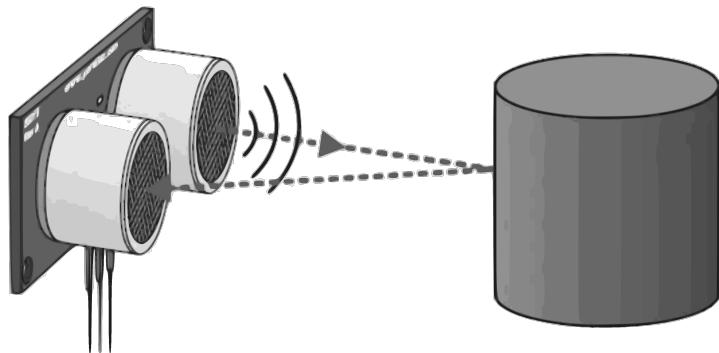


Figura 3.2: Funcionamento do sensor ultrassônico. Um alto-falante emite a onda sonora enquanto o outro recebe.

Os métodos de estimar distâncias baseados em tempo são conhecidos por *tempo de chegada* (ToA, de *Time-of-Arrival*) e *diferença de tempo de chegada* (TDoA, *Time-Difference-of-Arrival*) [26].

Pode-se utilizar a transmissão de dois sinais simultâneos com velocidades diferentes e conhecidas para estimar distâncias. É o que se faz, por exemplo, para saber a que distância um trovão caiu. Quando um trovão aparece, dois sinais com velocidades diferentes são enviados: um sinal luminoso (que viaja a velocidade da luz) e um sinal sonoro (que viaja a velocidade do som). Medindo a diferença entre o recebimento dos dois sinais, faz-se a estimativa da distância (veja a Figura 3.3).

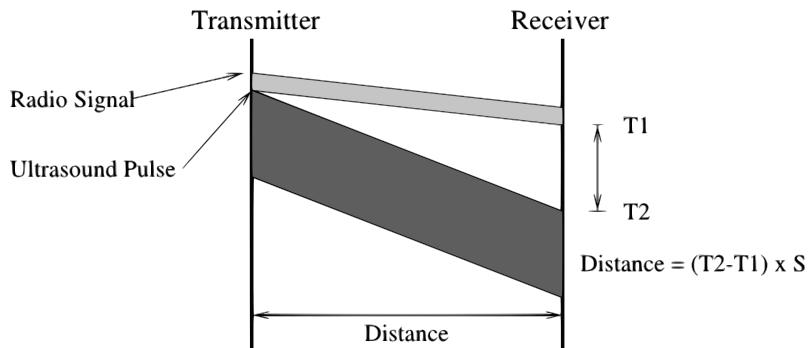


Figura 3.3: Estimativa de distância usando a diferença entre um sinal ultrassônico e um de rádio [26].

Savvides *et al.* [26] construíram outro protótipo, chamado Medusa, utilizando deste princípio. Para grandes distâncias, infelizmente, a utilização de sensores ultrassônicos mostrou-se ser um problema [40]. Porém, para pequenas distâncias os resultados foram promissores, com precisões de dois centímetros para sensores separados por três metros.

Em curtas distâncias, por tanto, utilizar métodos baseados em atraso do tempo de recebimento mostrou-se uma boa alternativa frente aos métodos de medição da intensidade do sinal.

3.1.3 Sobre o conjunto de sensores

Tendo estabelecida uma forma de obter as distâncias, como discutido na seção 13, precisa-se definir um conjunto de sensores como âncoras para poder calcular a rea-lização dos demais sensores. Afim de aplicar o Algorítimo 2, caso o problema seja definido em \mathbb{R}^2 ou \mathbb{R}^3 , precisa-se de 3 e 4 sensores como âncoras, respectivamente. Em [43], Fekete e Jordán apresentam um estudo sobre geometrias mínimas para as âncoras que continuam garantindo a unicidade da solução.

Além disso, também deve-se garantir que estes sensores estejam próximos o suficiente para que se possa estimar distâncias (garantindo a existência de $K+1$ -cliques). Por isso, deve-se pensar em um conjunto de restrições no posicionamento das âncoras e movimento dos robôs móveis de forma a garantir que sempre haverá uma ordem de K -lateração. Eren *et al.* [2] apresenta um estudo sobre a realização de grafos com geometrias aleatórias.

Afim de ilustrar e averiguar a eficiência do que foi apresentado aqui, realizou-se algumas simulações computacionais implementando os Algorítimos 1 e 2 em C.

Tais simulações foram executadas em um computador pessoal utilizando o sistema operacional Manjaro (uma distribuição Linux baseada em Arch), equipado com um processador Intel Core i5-8600K (6 núcleos operando em 4.1Ghz) e um pente de memória DDR4 2666MHz de 8Gb em *Single-Channel* (operando, por tanto, em 1333Mhz).

Para a eficiência dos algorítimos, fez-se uso da biblioteca de código aberto LAPACKE, que é uma interface para o LAPACK (*Linear Algebra Packge*) em C, onde encontra-se algumas implementações muito otimizadas de operações elementares envolvendo Álgebra Linear. Por exemplo, possui funções para solucionar sistemas matriz-vetor $Ax = b$, através da decomposição matricial LU [35], que pôde ser usada para solucionar o sistema linear 2.11, discutido na seção 2.2.5.

3.2 Medindo erros

Como tanto o algorítimo 1 quanto o 2 calculam posições em função de realizações previamente calculadas, é inevitável o acumulo de erros entre essas realizações. Mesmo que a solução teórica seja perfeita, na prática os valores não são representados exatamente. Isso se dá pois o computador não trabalha no conjunto dos números reais, ou, pior, para qualquer $i \in \mathbb{R}$, a possibilidade de i não pode ser representado por um computador tem probabilidade 1. Por isso, gera-se a necessidade de analisar o quanto perto as soluções calculadas estão das reais.

Neste texto, a principal medida de confiabilidade de uma solução será calculada através da *Mean Distance Error*, como segue [44].

Mean Distance Error (MDE): Seja $G = (V, E, d)$ um grafo ponderado que defina uma instância DGP. Se $\{x_1, \dots, x_n\}$ é um conjunto que define uma realização dos n vértices de G e então,

$$MDE(x) = \frac{1}{|d|} \sum_{i,j} \frac{|||x_i - x_j|| - d_{i,j}|}{d_{i,j}}.$$

3.3 Exemplares utilizados

Como entrada das simulações, implementou-se o Algorítimo 3, que gera um grafo K -laterativo com n vértices de coordenadas aleatórias $x_i = (x_{i1}, \dots, x_{iK}) \in \mathbb{R}^K$ e provê um conjunto limitável de arestas. Esse limitação é definida pelo parâmetro $\mathcal{P} \in [0, 1]$, como coeficiente de probabilidade de uma aresta ser ou não acessível entre dois vértices aleatório v_i e v_j , com $i, j > K + 1$ (garantindo que sempre haverá ao menos uma $(K+1)$ -clique adjacente a todo vértice).

Algorithm 3: $G = \text{criaInstancia}(n, \mathcal{P})$

```

// Percorra todos os vértices
1 for  $i \in \{0, \dots, n\}$  do
    // Varie entre cada coordenada de  $x_i \in \mathbb{R}^K$ 
    2   for  $j \in 1, \dots, K$  do
        /* Atribua um valor aleatório para cada co. A função
           Aleatorio( $m$ ) gera valores entre 0 e  $m$ . */
        3      $x_{ij} = \text{Aleatorio}(m);$ 
    4   end
5 end
    // Para cada vértice, percorra todos os seus antecessores
6 for  $x_i \in \{x_n, \dots, x_1\}$  do
7   for  $x_j \in \{x_1, \dots, x_{i-1}\}$  do
        // Gere um número  $a \in [0, 1]$  aleatório.
        8       Seja  $a = \text{Aleatorio}(1);$ 
        // Verifique se  $a > \mathcal{P}$  e se  $i, j > K$ 
        9       if  $a > \mathcal{P}$  and  $i > K$  and  $j > K$  then
            // Defina uma aresta entre  $x_i$  e  $x_j$ 
        10       $e_{i,j} = \|x_i - x_j\|;$ 
        11    end
    12  end
13 end
14 return  $G = (\{v_1, \dots, v_n\}, \{\{e_{1,2}\}, \dots, \{e_j\}\});$ 

```

Por exemplo, com o Algorítimo 3 gerou-se a instância mostrada na Figura 3.4.

x	y	z	
(33, 36, 27)			0.000000 20.904545 23.000000 26.832816 31.208973
(15, 43, 35)			20.904545 0.000000 25.258662 32.572995 47.592016
(36, 42, 49)			23.000000 25.258662 0.000000 40.112342 49.000000
(21, 12, 27)			26.832816 32.572995 40.112342 0.000000 23.790755
(40, 9, 13)			31.208973 47.592016 49.000000 23.790755 0.000000

Figura 3.4: A esquerda, as coordenadas dos vértices gerados e, a direita, a matriz de distância a eles associados.

3.4 Simulações

Primeiramente, para gerar instâncias do algoritmo 1, deve-se usar o algoritmo 3 com $\mathcal{P} = 100$. Assim pode-se garantir a geração de um grafo completo. Criou-se, então, instâncias com tamanhos variados: $n \in \{5, 6, 7, 10, 15, 20, 30, 40, 50, 100, 200, 400, 500\}$. A seguir apresenta-se alguns gráficos sobre os resultados.

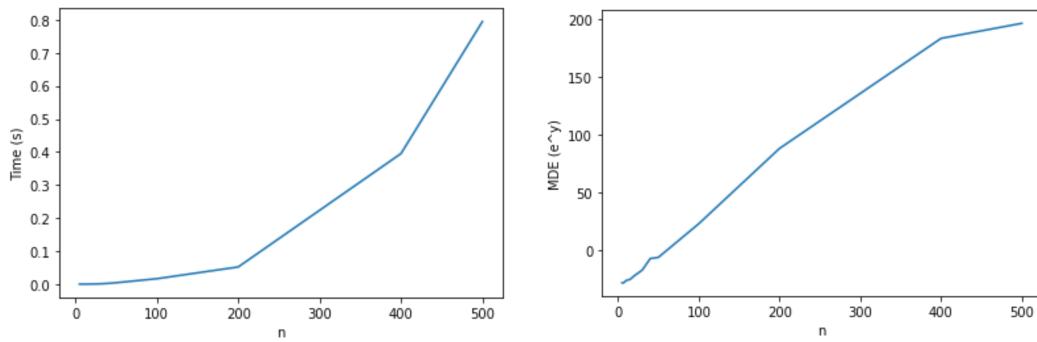


Figura 3.5: A esquerda, o tempo de processo e, a direita, a ordem de grandeza associada ao MDE das soluções.

Perceba o comportamento mostrado na Figura 3.5. Teve-se que fazer uma linearização no gráfico que representa o MDE da solução, pois este cresceu exponencialmente e foi para a ordem de 10^{200} em 500 vértices. Esse comportamento se deu por conta dos erros acumulados entre as iterações.

Uma proposta pensada para contornar essa situação infeliz foi alterar o valor de \mathcal{P} , deixando o menor, obrigando o algoritmo a calcular realizações utilizando vértices iniciais. Porém, isso implicaria em um grafo não completo, o que não condiz definição do Algoritmo 1.

Outra solução proposta fora de utilizar sempre os primeiros vértices no lugar dos antecessores mais próximos. Isso significaria utilizar sempre os vértices âncoras para realizar os demais, semelhante com o que é feito no sistema GPS. Com isso, tivemos os resultados satisfatórios apresentados pela Figura 3.6. Perceba que mesmo com 500 vértices ainda obteve-se resultados com MDE na ordem de 10^{-20} .

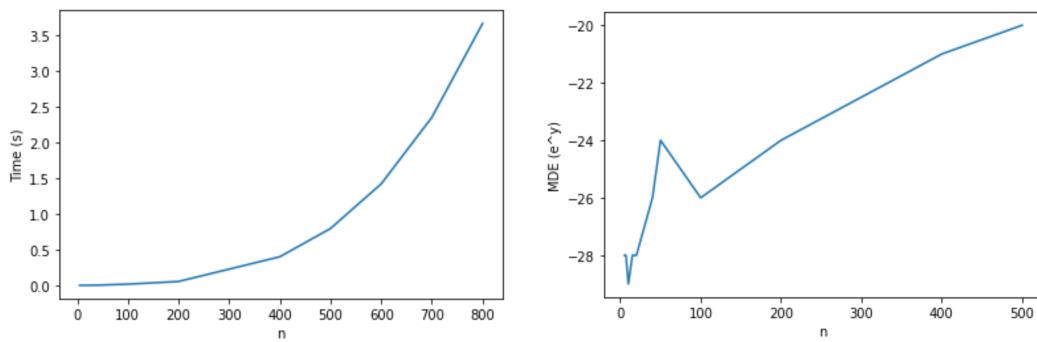


Figura 3.6: A esquerda, o tempo de processo e, a direita, a ordem de grandeza associada ao MDE das soluções.

Também simulou-se o Algorítimo 2, que teve um comportamento muito similar ao Algorítimo 1, como era de se esperar. Também usou-se instâncias n variando em $\{5, 6, 7, 10, 15, 20, 30, 40, 50, 100, 200, 400, 500\}$. Porém, agora, com uma busca inteligente por vértices adjacentes, pode-se variar \mathcal{P} afim de analisar diferentes geometrias para o problema, como apresenta-se a seguir.

Instâncias com 70% de arestas

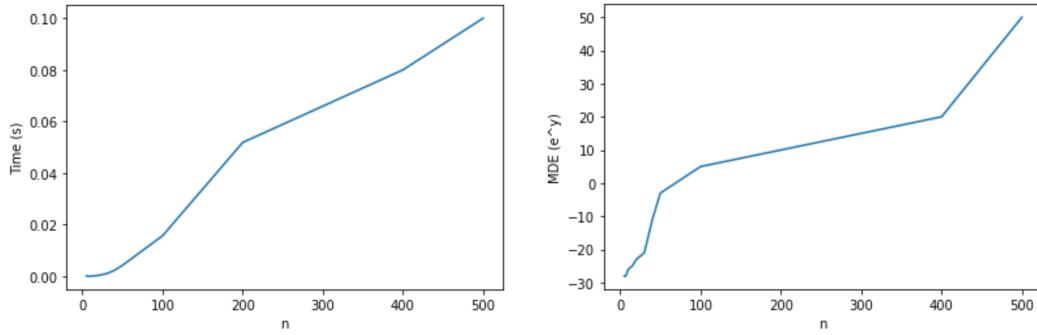


Figura 3.7: A esquerda, o tempo de processo e, a direita, a ordem de grandeza associada ao MDE das soluções. Utilizou-se $\mathcal{P} = 0.7$.

Instâncias com 5% de arestas

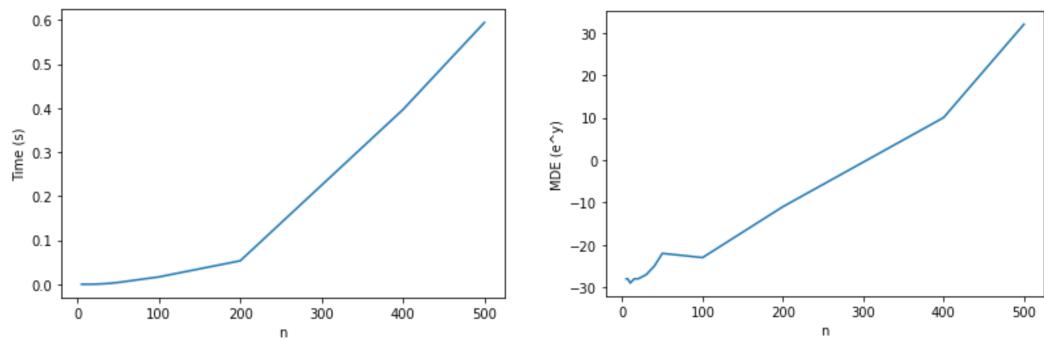


Figura 3.8: A esquerda, o tempo de processo e, a direita, a ordem de grandeza associada ao MDE das soluções. Utilizou-se $\mathcal{P} = 0.05$.

Como esperado, é perceptível uma grande melhora na MDE das soluções conforme se diminui o valor de \mathcal{P} .

Também deve-se perceber que o tempo não obedeceu um padrão perfeito. Podemos relacionar esse fato com a aleatoriedade das instâncias, como pode-se notar que o Algoritmo 2 teve maiores volatilidades no padrão temporal. Mesmo assim, há um consenso. As soluções mostradas nas Figuras 3.6 e 3.8, por exemplo, demonstram claramente um padrão polinomial, indo de encontro com o que foi apresentado na seção 13.

Capítulo 4

Considerações Finais

Com isso, concluí-se o estudo sobre a Geometria de Distâncias aplicada ao problema de localização de sensores, tal qual teve como resultado um algorítimo que garante encontrar a solução do problema (se houver). .

Nos cabe, nesse momento, voltarmos atenção às propostas levantadas internamente no início do projeto e verificar se elas foram cumpridas. Seguem o conjunto de objetivos específicos desse projeto, munidos de breve conclusão:

1. Estudar formas viáveis (pelos vieses energético, de construção e precisão da medida) para obtenção das distâncias entre os elementos do sistema físico:

Fortemente embasado em [26] e [40], faz-se uma apresentação sobre esses conceitos na Seção 3.1.2.

2. Estudar as possíveis distribuições dos Robôs Móveis em um sistema genérico, visando verificar quais conjuntos de dados possam ser garantidos como entradas para a construção do problema:

Além de estudar distribuições genéricas, como em [2], desenvolve-se na Seção 3 o Algorítimo 3, que cria instâncias aleatórias com diferentes densidades de ligações.

3. Verificar a solução do Discretizable Order Problem [4] aplicado ao problema proposto e estudar o ordenamento de vértices que se adeque aos objetivos do trabalho:

Durante o desenvolvimento da Seção 2.2, percebeu-se que uma ordem de discretização não seria necessária, visto que a definição do problema necessita de uma única solução [12]. Por conta disso, definiu-se a ordem de K -lateração, que garante a unicidade de solução.

4. Caso consiga-se uma boa ordenação para os vértices, verificar a aplicação do algorítimo Branch-And-Prune [45] para a solução do problema proposto. Se não for possível, estudar outros algorítimos que possam solucionar o problema:

Visto que o problema não era discreto, não fora possível aplicar o algorítimo Branch-And-Prune. Ao invés disso, se propôs o Algorítimo 2, que utiliza a trilateração como núcleo e garante no máximo uma solução.

5. Estudar a complexidade computacional do algoritmo proposto aplicado as possíveis distribuições:

Isso foi feito no fim das Seções 2.2 e 3, onde verificou-se que o algoritmo pode ser resolvido em tempo polinomial. Também mostrou-se um estudo sobre diferentes visões da geometria do problema afim de minimizar os erros associados.

6. Simular computacionalmente o algoritmo para solução do problema com instâncias artificialmente geradas, dominando cada passo utilizado:

Simulações apresentadas, de forma satisfatória, na Seção 3.

7. Aplicar o algoritmo estudado em estruturas de pequena escala, como instâncias reais do problema:

Não fora possível cumprir este último objetivo. Infelizmente, a manufatura de robôs móveis usando sensores de distância se demonstrou excessivamente custosa [26, 40].

Vale lembrar, porém, que a utilização de instâncias artificiais genéricas se mostrou de grande importância. Pode-se averiguar e discutir sobre diferentes geometrias, mesmo sem a prototipação de fato.

Como experiências futuras, deseja-se estudar mais sobre a obtenção e tratamento de dados de distâncias, visto que esses possuem erros de medida [40] (não levados em consideração nesse trabalho). Alguns trabalhos relacionados à Geometria de Distâncias aplicada ao caso intervalar são mostrados em [46]. Há curiosidade em ver como medida MDE se comporta para uma quantidade grande de vértices associados a erros de medida.

Também deseja-se fazer estudos sobre geometrias mínimas para o problema, como feito em [43]. Deseja-se verificar um conjunto de condições que restringem o movimento de um robô móvel afim de garantir que este ainda possa ser localizado.

Referências Bibliográficas

- [1] Gerald Cook and Feitian Zhang. *Mobile Robots: Navigation, Control and Sensing, Surface Robots and AUVs*. John Wiley & Sons, 2020.
- [2] Tolga Eren, OK Goldenberg, Walter Whiteley, Yang Richard Yang, A Stephen Morse, Brian DO Anderson, and Peter N Belhumeur. Rigidity, computation, and randomization in network localization. In *IEEE INFOCOM 2004*, volume 4, pages 2673–2684. IEEE, 2004.
- [3] Spyros G Tzafestas. *Introduction to mobile robot control*. Elsevier, 2013.
- [4] Leo Liberti, Carlile Lavor, Nelson Maculan, and Antonio Mucherino. Euclidean distance geometry and applications. *Society for Industrial and Applied Mathematics*, 56(1):3–69, February 2014.
- [5] J. A. Bondy and U. S. R. Murty. *Graph Theory With Applications*. Elsevier Science Publishing, New York, 5 edition, 1982.
- [6] Leonhard Euler. Leonhard euler and the königsberg bridges. *Scientific American*, 189(1):66–72, 1953.
- [7] W.W. Rouse Ball. Hsm coxeter mathematical recreations and essays, 1956.
- [8] Frank Harary. *Graph Theory*. Westview Press, 1969.
- [9] Gustav Kirchhoff. Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird. *Annalen der Physik*, 148(12):497–508, 1847.
- [10] A Cayley. On the theory of the analytical forms called trees, math, 1897.
- [11] Jayme Luiz Szwarcfiter. *Teoria computacional de grafos: Os algoritmos*. Elsevier Brasil, 2018.
- [12] Leo Liberti and Carlile Lavor. *Euclidean Distance Geometry*. Springer, 2017.
- [13] Irineu Bicudo et al. *Os elementos*. Unesp, 2009.
- [14] Arthur Cayley. A theorem in the geometry of position. *Cambridge Mathematical Journal*, 2:267–271, 1841.
- [15] Karl Menger. Untersuchungen über allgemeine metrik. *Mathematische Annalen*, 100(1):75–163, 1928.

- [16] Leonard M Blumenthal. *Theory and applications of distance geometry*. Oxford University Press, Oxford, 1953.
- [17] Manfred J Sippl and Harold A Scheraga. Cayley-menger coordinates. *Proceedings of the National Academy of Sciences*, 83(8):2283–2287, 1986.
- [18] Yechiam Yemini. Some theoretical aspects of position-location problems. In *20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*, pages 1–8. IEEE, 1979.
- [19] Gordon M Crippen, Timothy F Havel, et al. *Distance geometry and molecular conformation*, volume 74. Research Studies Press Taunton, 1988.
- [20] Yechiam Yemini. The positioning problem-a draft of an intermediate summary. Technical report, UNIVERSITY OF SOUTHERN CALIFORNIA MARINA DEL REY INFORMATION SCIENCES INST, 1978.
- [21] Deepak Tolani, Ambarish Goswami, and Norman I Badler. Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical models*, 62(5):353–388, 2000.
- [22] Jan de Leeuw and Willem Heiser. 13 theory of multidimensional scaling. *Handbook of statistics*, 2:285–316, 1982.
- [23] David L Nelson and Michael M Cox. *Lehninger principles of biochemistry*. W.H.Freeman and Company, 2013.
- [24] Carlile Lavor. *Uma abordagem determinística para minimização global da energia potencial de moléculas*. PhD thesis, PhD thesis, COPPE/UFRJ, Rio de Janeiro, 2001.
- [25] GN Ramachandran, AS Kolaskar, C Ramakrishnan, and V Sasisekharan. The mean geometry of the peptide unit from crystal structure data. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 359(2):298–302, 1974.
- [26] Andreas Savvides, Chih-Chieh Han, and Mani B Srivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 166–179. ACM, 2001.
- [27] C. Lavor, N. Maculan, M. Souza, and R. Alves. *Álgebra e Geometria no Cálculo de Estrutura Molecular*. IMPA, Rio de Janeiro, RJ, 31º colóquio brasileiro de matemática edition, 2017.
- [28] James B Saxe. Embeddability of weighted graphs in k-space is strongly np-hard. In *Proc. of 17th Allerton Conference in Communications, Control and Computing, Monticello, IL*, pages 480–489, 1979.
- [29] Carlile Lavor, Leo Liberti, Weldon A Lodwick, and Tiago Mendonça da Costa. *An Introduction to Distance Geometry applied to Molecular Geometry*. Springer, 2017.

- [30] Riccardo Benedetti and Jean-Jacques Risler. In real algebraic and semi-algebraic sets. *Berlin, Hermann, Paris*, 1990.
- [31] Bruce Hendrickson. Conditions for unique graph realizations. *SIAM journal on computing*, 21(1):65–84, 1992.
- [32] Robert Connelly. On generic global rigidity, applied geometry and discrete mathematics, 147–155. *DIMACS Ser. Discrete Math. Theoret. Comput. Sci*, 4, 1991.
- [33] Abdo Y Alfakih. *Euclidean distance matrices and their applications in rigidity theory*. Springer, 2018.
- [34] Qunfeng Dong and Zhijun Wu. A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. *Journal of Global Optimization*, 22(1-4):365–375, 2002.
- [35] Elon Lages Lima. *Álgebra Linear*. SBM, Rio de Janeiro : IMPA, 1a edition, 2014.
- [36] Bernhard Hofmann-Wellenhof, Herbert Lichtenegger, and James Collins. *Global positioning system: theory and practice*. Springer Science & Business Media, 2012.
- [37] Ana Flávia da Cunha Lima. Rigidez de grafos e aplicações. Master’s thesis, UNICAMP, IMECC, Campinas, SP, 2015.
- [38] Gregor Klancar, Andrej Zdesar, Saso Blazic, and Igor Skrjanc. *Wheeled mobile robotics: from fundamentals towards autonomous systems*. Butterworth-Heinemann, 2017.
- [39] James F Campbell, Don Sweeney, and Juan Zhang. Strategic design for delivery with trucks and drones. *Supply Chain Analytics Report SCMA (04 2017)*, 2017.
- [40] HR Everett. *Sensors for mobile robots*. CRC Press, 1995.
- [41] Eduardo Bayro-Corrochano. *Geometric computing: for wavelet transforms, robot vision, learning, control and action*. Springer Publishing Company, Incorporated, 2010.
- [42] Herch Moysés Nussenzveig. *Curso de física básica: Eletromagnetismo (vol. 3)*, volume 3. Editora Blucher, 2015.
- [43] Zsolt Fekete and Tibor Jordán. Uniquely localizable networks with few anchors. In *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*, pages 176–183. Springer, 2006.
- [44] Antonio Mucherino, Leo Liberti, and Carlile Lavor. Md-jeep: an implementation of a branch and prune algorithm for distance geometry problems. In *International Congress on Mathematical Software*, pages 186–197. Springer, 2010.

- [45] Leo Liberti, Carlile Lavor, and Nelson Maculan. A branch-and-prune algorithm for the molecular distance geometry problem. *International Transactions in Operational Research*, 15(1):1–17, 2008.
- [46] Anthony Man-Cho So and Yinyu Ye. Theory of semidefinite programming for sensor network localization. *Mathematical Programming*, 109(2-3):367–384, 2007.

Apêndice A

Métricas

Como esse texto utiliza fortemente o conceito de distância, é necessário e bem vindo que se gaste algum espaço para uma construção formal dessa ideia. A noção de distância está relacionada com o conceito de *métrica*, como segue.

Seja \mathcal{X} um espaço vetorial K -dimensional sobre \mathbb{R} . *Métrica* é uma função de dois argumentos que mapeia pares ordenados de elementos em \mathcal{X} para um número real não negativo. Precisamente, para todo $x, y \in \mathcal{X}$, uma função $d(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ é uma métrica se satisfaz os seguintes axiomas:

1. $d(x, y) = 0$ se, e somente se, $x = y$;
2. $d(x, y) = d(y, x)$;
3. $d(x, z) \leq d(x, y) + d(y, z)$;
4. $d(x, y) \geq 0$

Nesse trabalho, quando não é especificado qual métrica se está usando, fica implícita a utilização da *Métrica Euclidiana*, definida em função da *Norma Euclidiana*:

$$\forall x, y \in \mathcal{X}, d(x, y) = \|x - y\|_2 = \sqrt{\langle x, y \rangle} = \sqrt{\sum_{i=1}^K (x_i - y_i)^2}. \quad (\text{Norma Euclidiana})$$

O par (\mathcal{X}, d) é chamado *espaço métrico*. A noção de métrica não depende de espaços vetoriais, donde pode ser facilmente generalizada fazendo \mathcal{X} um conjunto qualquer.

Apêndice B

Rigidez

O conceito de rigidez em grafos é pouco simples e suficientemente extenso para que não nos caiba abordá-lo completamente nesse texto. Em contrapartida, visto que o objeto central desse texto (soluções de instâncias WSNL) tem estreita relação com a definição de *rigidez global*, no que segue, apresentar-se-á satisfatoriamente esta definição.

Para um maior aprofundamento, recomenda-se [12, 37].

O conceito de *rigidez* está associado com a ideia de *movimento rígido* entre as possíveis realizações de um grafo.

Definição: Um *framework* \mathcal{F} em \mathbb{R}^K é um par (G, x) , onde x é uma realização de G em \mathbb{R}^K .

Definição: Um *movimento do framework* (G, x) é uma função contínua $y : [0, 1] \rightarrow \mathbb{R}^{nK}$, tal que:

- $y(0) = x$;
- $y(t)$ é uma realização válida de G para todo $t \in [0, 1]$.

Definição: Duas realizações x, y de um grafo $G = (V, E)$ são ditos *congruentes* se, para todo $u, v \in V$, tem-se $\|x_u - x_v\| = \|y_u - y_v\|$. Se x, y não são congruentes, então eles são *incongruentes*.

Definição: Uma *flexão* de um framework (G, x) é um deslocamento y de x tal que $y(t)$ é incongruente a x para qualquer $t \in (0, 1]$.

Definição: Um framework é *flexível* se possuir uma flexão; do contrário, é dito *rígido*.

Definição: Dois frameworks $\mathcal{F} = (V, E, x)$ e $\mathcal{F}' = (V', E', x')$ são *equivalentes* em \mathbb{R}^K se há uma bijeção entre os conjuntos de vértices V e V' que preservem os comprimentos das arestas associados a esses.

Definição: Um framework $\mathcal{F} = (G, x)$ é *globalmente rígido* em \mathbb{R}^K se, quando \mathcal{F} é equivalente a $\mathcal{F}' = (G, x')$, então x é congruente a x' .