Universidade do Estado do Rio de Janeiro
Centro de Tecnologia e Ciências
Faculdade de Engenharia

Alan Oliveira de Sá

Localização Colaborativa em Robótica de Enxame

Rio de Janeiro
2015

Alan Oliveira de Sá

Localização Colaborativa em Robótica de Enxame

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientadora: Prof.ª Dr.ª Nadia Nedjah
Orientadora: Prof.ª Dr.ª Luiza de Macedo Mourelle

Rio de Janeiro
2015

CATALOGAÇÃO NA FONTE
UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

S111 Sá, Alan Oliveira de
Localização Colaborativa em Robótica de Enxame/Alan Oliveira de Sá. – 2015.
125 f.

Orientadora: Nadia Nedjah.
Orientadora: Luiza de Macedo Mourelle.
Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Engenharia Eletrônica. 2. Sistemas inteligentes de controle - Dissertações. 3. Robotica - Dissertações. 4. Inteligência coletiva - Dissertações. I. Nedjah, Nadia. II. Mourelle, Luiza de Macedo. III. Universidade do Estado do Rio de Janeiro. IV. Título.

CDU 007.52

Autorizo, apenas para fins acadêmicos e científicos, a reprodução total ou parcial desta dissertação, desde que citada a fonte.

Assinatura

Data

Alan Oliveira de Sá

Localização Colaborativa em Robótica de Enxame

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em: 26 de maio de 2015.

Banca Examinadora:

Prof.ª Dr.ª Nadia Nedjah (Orientadora)
Faculdade de Engenharia, UERJ

Prof.ª Dr.ª Luiza de Macedo Mourelle (Orientadora)
Faculdade de Engenharia, UERJ

Prof. Dr. Valmir Carneiro Barbosa
Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ

Prof. Dr. José Manoel de Seixas
Programa de Engenharia Elétrica, COPPE/UFRJ

Rio de Janeiro
2015

DEDICATÓRIA

Dedico este trabalho à minha filha Beatriz e à minha esposa Adriana, as quais me alimentaram com amor e carinho em todos os momentos desta jornada, mesmo quando, em virtude do estudo, não pude retribuir na mesma proporção. Vocês são a minha vida e a minha motivação! Aos meus pais pelo exemplo, pela inestimável ajuda, pelas palavras de incentivo e pela minha formação em todos os sentidos. Ao meu irmão, Ruy, meu grande amigo, e à minha avó Beliza que sempre torcem por minhas conquistas. Aos meus sogros pelo apreço e por todo o apoio prestado neste período.

A diferença entre o desenhista e o artista, é que o desenhista percebe e reproduz um conjunto de retas e curvas. Já o artista, lê o sentido das retas e curvas e soma à elas o seu sentimento.

RESUMO

SÁ, Alan Oliveira de. *Localização Colaborativa em Robótica de Enxame*. 2015. 125f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2015.

Diversas das possíveis aplicações da robótica de enxame demandam que cada robô seja capaz de estimar a sua posição. A informação de localização dos robôs é necessária, por exemplo, para que cada elemento do enxame possa se posicionar dentro de uma formatura de robôs pré-definida. Da mesma forma, quando os robôs atuam como sensores móveis, a informação de posição é necessária para que seja possível identificar o local dos eventos medidos. Em virtude do tamanho, custo e energia dos dispositivos, bem como limitações impostas pelo ambiente de operação, a solução mais evidente, *i.e.* utilizar um Sistema de Posicionamento Global (GPS), torna-se muitas vezes inviável. O método proposto neste trabalho permite que as posições absolutas de um conjunto de nós desconhecidos sejam estimadas, com base nas coordenadas de um conjunto de nós de referência e nas medidas de distância tomadas entre os nós da rede. A solução é obtida por meio de uma estratégia de processamento distribuído, onde cada nó desconhecido estima sua própria posição e ajuda os seus vizinhos a calcular as suas respectivas coordenadas. A solução conta com um novo método denominado *Multi-hop Collaborative Min-Max Localization* (MCMM), ora proposto com o objetivo de melhorar a qualidade da posição inicial dos nós desconhecidos em caso de falhas durante o reconhecimento dos nós de referência. O refinamento das posições é feito com base nos algoritmos de busca por retrocesso (BSA) e de otimização por enxame de partículas (PSO), cujos desempenhos são comparados. Para compor a função objetivo, é introduzido um novo método para o cálculo do fator de confiança dos nós da rede, o Fator de Confiança pela Área Min-Max (MMA-CF), o qual é comparado com o Fator de Confiança por Saltos às Referências (HTA-CF), previamente existente. Com base no método de localização proposto, foram desenvolvidos quatro algoritmos, os quais são avaliados por meio de simulações realizadas no MATLAB® e experimentos conduzidos em enxames de robôs do tipo *Kilobot*. O desempenho dos algoritmos é avaliado em problemas com diferentes topologias, quantidades de nós e proporção de nós de referência. O desempenho dos algoritmos é também comparado com o de outros algoritmos de localização, tendo apresentado resultados 40% a 51% melhores. Os resultados das simulações e dos experimentos demonstram a eficácia do método proposto.

Palavras-chave: Localização; Robótica de Enxame; Inteligência Coletiva; Rede de Sensores sem Fio; Otimização por Enxame de Partículas; Algoritmo de Busca por Retrocesso.

ABSTRACT

SÁ, Alan Oliveira de. *Collaborative Localization in Swarm Robotics*. 2015. 125f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2015.

Many applications of Swarm Robotic Systems (SRSs) require that a robot is able to discover its position. The location information of the robots is required, for example, to allow them to be correctly positioned within a predefined swarm formation. Similarly, when the robots act as mobile sensors, the position information is needed to allow the identification of the location of the measured events. Due to the size, cost and energy source restrictions of these devices, or even limitations imposed by the operating environment, the straightforward solution, *i.e.* the use of a Global Positioning System (GPS), is often not feasible. The method proposed in this work allows the estimation of the absolute positions of a set of unknown nodes, based on the coordinates of a set of reference nodes and the distances measured between nodes. The solution is achieved by means of a distributed processing strategy, where each unknown node estimates its own position and helps its neighbors to compute their respective coordinates. The solution makes use of a new method called Multi-hop Collaborative Min-Max Localization (MCMM), herein proposed, aiming to improve the quality of the initial positions estimated by the unknown nodes in case of failure during the recognition of the reference nodes. The positions refinement is achieved based on the Backtracking Search Optimization Algorithm (BSA) and the Particle Swarm Optimization (PSO), whose performances are compared. To compose the objective function, a new method to compute the confidence factor of the network nodes is introduced, the Min-max Area Confidence Factor (MMA-CF), which is compared with the existing Hops to Anchor Confidence Factor (HTA-CF). Based on the proposed localization method, four algorithms were developed and further evaluated through a set of simulations in MATLAB® and experiments in swarms of type *Kilobot* robots. The performance of the algorithms is evaluated on problems with different topologies, quantities of nodes and proportion of reference nodes. The performance of the algorithms is also compared with the performance of other localization algorithms, showing improvements between 40% to 51%. The simulations and experiments outcomes demonstrate the effectiveness of the proposed method.

Keywords: Localization; Swarm Robotic System; Swarm Intelligence; Wireless Sensor Network; Particle Swarm Optimization; Backtracking Search Optimization Algorithm.

LISTA DE FIGURAS

LISTA DE FIGURAS

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

SUMÁRIO

SUMÁRIO

INTRODUÇÃO

EM um conjunto de robôs, também denominado enxame de robôs, ou SRS (*Swarm Robotic System*), a capacidade do coletivo é muitas vezes igual ou maior do que a soma das capacidades de cada robô que compõe tal coletividade (RUBENSTEIN, 2009). Esta capacidade adicional, alcançada pelo coletivo, é consequência da soma dos esforços do enxame, concentrada em torno da consecução de um objetivo comum.

Em geral, em um enxame de robôs, cada robô é dotado de recursos sensoriais, motores e de comunicação limitados, porém suficientes para que cada indivíduo seja capaz de executar tarefas simples. Os sensores permitem que cada indivíduo seja capaz de perceber e extrair informações do ambiente, tais como a intensidade luminosa (RUBENSTEIN; AHLER; NAGPAL, 2012), distância para obstáculos (MONDADA et al., 2009; GCTRONIC, 2015), som (MONDADA et al., 2009), ou mesmo imagens (MONDADA et al., 2009). Os recursos motores permitem que cada robô se movimente pelo ambiente, ampliando sua capacidade exploratória, bem como permitindo que o mesmo se posicione dentro de uma formatura de robôs. A movimentação pode ser obtida, por exemplo, por meio de rodas (MONDADA et al., 2009; GCTRONIC, 2015), motores vibratórios (RUBENSTEIN; AHLER; NAGPAL, 2012) ou hélices (LEHNERT; CORKE, 2013). Os sistemas de comunicação dos robôs permitem que os mesmos compartilhem as informações adquiridas com outros indivíduos do enxame, perfazendo um recurso fundamental para o estabelecimento de comportamentos coletivos. Na robótica de enxame, a transmissão de dados ocorre, por exemplo, por meio de sinais de infravermelho (RUBENSTEIN; AHLER; NAGPAL, 2012) ou de RF (Radiofrequência) (MONDADA et al., 2009; LEHNERT; CORKE, 2013; GCTRONIC, 2015). São ainda atributos típicos dos robôs de um SRS: o tamanho reduzido, a limitada fonte de energia e o baixo custo.

Nos SRSs, as características de escalabilidade e robustez são desejáveis e têm motivado as pesquisas acerca destes sistemas. A escalabilidade corresponde à capacidade que o SRS tem de operar com diferentes quantidades de indivíduos, de forma que a sua operação não seja afetada por alterações no tamanho do enxame (ŞAHIN, 2005). Graças

à escalabilidade é possível, muitas vezes, aumentar o desempenho do sistema pela simples adição de novos indivíduos ao enxame. A robustez exige que, em caso de falha de algum robô, o SRS seja capaz de continuar operando, mesmo que com um menor desempenho (ŞAHIN, 2005).

Em (ŞAHIN, 2005), são elencados alguns tipos de tarefas e problemas reais aos quais os SRS poderiam ser aplicados:

- Tarefas perigosas: diferente do que ocorre em soluções que utilizam apenas um robô complexo, os robôs de um SRS podem ser considerados descartáveis, devido ao custo relativamente baixo. Isto torna os SRS adequados para o uso em atividades que envolvam risco como, por exemplo, a abertura de caminhos em campos minados (ŞAHIN, 2005) ou o monitoramento de resíduos nucleares armazenados debaixo d'água (NAWAZ et al., 2010).

- Tarefas que possam aumentar ou diminuir com o tempo: graças à escalabilidade, os SRSs possuem a capacidade de aumentar ou diminuir de tamanho, conforme a necessidade da tarefa. Assim, em um vazamento de óleo no mar, por exemplo, robôs capazes de fazer a contenção ou o combate do derramamento do óleo podem ser adicionados ao sistema à medida que há um aumento do derramamento (ŞAHIN, 2005; KAKALIS; VENTIKOS, 2008), garantindo a eficácia da solução.

- Tarefas que requeiram redundância: a robustez de um SRSs resulta da redundância implícita no enxame. No caso de falhas inopinadas de alguns indivíduos, a redundância faz com que os indivíduos remanescentes absorvam suas tarefas, minimizando o impacto da falha. Por exemplo, em um campo de batalha, um SRS pode ser utilizado para constituir uma rede de comunicação dinâmica (ŞAHIN, 2005). Caso um robô seja abatido, a rede de comunicações pode ser recuperada pelo re-roteamento dos dados e/ou pelo reposicionamento físico dos robôs.

- Tarefas que necessitem cobrir uma região: os SRS são sistemas distribuídos e, portanto, são adequados para tarefas que se estendam por uma região. O monitoramento ambiental de lagos, rios e florestas (ŞAHIN, 2005; RODIC; KATIE; MESTER, 2009), por exemplo, constitui um bom domínio de aplicação para SRSs. A habilidade de sensoriamento distribuído dos SRSs pode ser uma valiosa ferramenta de detecção de eventos que impliquem em risco ambiental, como derramamentos de

resíduos tóxicos. A esse respeito, os SRSs possuem vantagens sobre as redes de sensores sem fio, ou WSN (*Wireless Sensor Network*), que podem ser consideradas como um SRS estático. Graças à mobilidade, os SRSs são capazes de se deslocar e mobilizar seus agentes em torno do foco do problema.

Sob alguns aspectos, as WSNs se assemelham aos SRSs, inclusive no que tange ao problema de localização de seus dispositivos, motivo pelo qual merecem ser abordadas nesta dissertação. De certa forma, podemos considerar uma WSN similar a um SRS estático (ŞAHIN, 2005).

As WSNs podem ser constituídas por diferentes tipos de sensores tais como sísmicos, magnéticos, térmicos, visuais, infravermelhos e acústicos (AKYILDIZ et al., 2002), que são capazes de monitorar uma ampla variedade de condições em diversos tipos de ambiente. Seus dispositivos elementares, os sensores, têm características similares aos robôs de um SRS. Assim como nos SRSs, os sensores de uma WSN possuem tamanho reduzido, limitada fonte de energia e devem ser de baixo custo. São dotados de recursos sensoriais, que permitem a obtenção de dados do ambiente, e de comunicação, que permitem o estabelecimento de redes para transmissão e roteamento dos dados coletados. Normalmente, as WSNs são compostas por centenas ou milhares de nós que são distribuídos pela área a ser monitorada (AKYILDIZ et al., 2002).

Em (ROMER; MATTERN, 2004), são listadas algumas aplicações concretas de WSNs, que avançaram além de uma mera visão e que estão bem documentadas. Tratam-se de WSNs para observação de aves em *Great Duck Island*, Maine, EUA; observação do comportamento animal dentro de um amplo habitat no *Mpala Research Center*, Quênia; monitoramento de ambientes subglaciais na geleira Briksdalsbreen, Noruega; criação de gado usando cercas virtuais; monitoramento do leito do mar na Inglaterra; monitoramento oceânico usando sensores à deriva, com o objetivo de obter uma cobertura global; monitoramento das condições que influenciam o crescimento de plantas, a fim de apoiar a agricultura de um vinhedo em Oregon, EUA; obtenção de dados para o resgate de vítimas de avalanches; monitoramento dos sinais vitais de pacientes em hospitais; monitoramento do consumo de energia em edifícios comerciais; rastreamento de veículos militares; e localização de atiradores e de trajetórias balísticas, com base na medição de sinais acústicos.

Diversas das possíveis aplicações de um SRS demandam que cada indivíduo seja capaz de descobrir sua posição. Da mesma forma as WSNs, na maioria dos casos, possuem

pouca utilidade quando não é possível identificar a posição de seus sensores (SUN; SU, 2011).

Nos SRSs, conforme apresentado em (RUBENSTEIN, 2009), a informação de localização dos robôs é necessária, por exemplo, para estabelecer comportamentos de autoformação, onde cada robô deve se posicionar dentro de uma organização pré-definida, ou mesmo comportamentos de recomposição, onde os robôs do enxame se reorganizam para reconstituir uma formação que tenha sido desfeita. Da mesma forma, quando um SRS atua como uma WSN móvel, a informação de posição é necessária para que seja possível localizar os eventos medidos. Em (NAWAZ et al., 2010), por exemplo, os robôs que monitoram os resíduos nucleares armazenados debaixo d'água precisam conhecer suas coordenadas para que a posição de um eventual vazamento possa ser identificada.

Nas WSNs, o conhecimento da localização dos nós é relevante não só para que as informações captadas por cada sensor sejam correlacionadas ao local em que foram obtidas, mas também para viabilizar o roteamento de dados por protocolos baseados na localização (AKKAYA; YOUNIS, 2005), tais como o MECN (*Minimum Energy Communication Network*) (RODOPLU; MENG, 1999), SMECN (*Small Minimum Energy Communication Network*) (LI; HALPERN, 2001), GAF (*Geographic Adaptive Fidelity*) (XU; HEIDEMANN; ESTRIN, 2001) e GEAR (*Geographic and energy-aware routing*) (YU; GOVINDAN; ESTRIN, 2001). Nestes protocolos de rotamento, a informação de localização é utilizada para encontrar o caminho que proporcione o menor consumo de energia de transmissão, permitindo, também, o direcionamento das requisições de dados à uma região específica. Por exemplo, se a região a ser medida é conhecida, a consulta pode ser difundida apenas para essa região em particular, com base na informação de localização de sensores, eliminando transmissões para regiões desnecessárias da rede (AKKAYA; YOUNIS, 2005).

Em ambos os casos supracitados, de SRSs e WSNs, o tamanho reduzido, a limitada fonte de energia e o baixo custo são características comuns dos dispositivos, sejam eles robôs ou sensores. Sendo assim, a solução mais evidente para o problema de localização, que consiste em equipar cada elemento com um receptor do Sistema de Posicionamento Global ou GPS (*Global Positioning System*), torna-se muitas vezes inviável. Adicionalmente, o uso de SRSs e WSNs em ambientes internos, ou mesmo subaquáticos, impede o recebimento de sinais de localização do GPS, inviabilizando este tipo de solução nos referidos ambientes. Por estes motivos, algumas soluções alternativas ao GPS têm sido

desenvolvidas para a localização dos dispositivos, ou nós, dos SRSs e das WSNs.

Nesta dissertação, é proposto um método colaborativo, onde os nós desconhecidos, *i.e.* nós que desconhecem suas coordenadas, se localizam com base nas informações de posição de um conjunto de nós referência, cujas coordenadas são conhecidas, e em medidas de distância entre os nós da rede. O método é dividido em quatro etapas. Na ETAPA-I, utiliza-se o algoritmo Sum-Dist (SAVVIDES; PARK; SRIVASTAVA, 2002), para medição de distâncias em múltiplos saltos. Na ETAPA-II, é utilizado o método Min-Max (SAVVIDES; PARK; SRIVASTAVA, 2002) para fazer uma estimativa inicial das posições dos robôs. A ETAPA-III visa melhorar a qualidade da posição inicial dos nós em caso de falhas durante o reconhecimento dos nós de referência na ETAPA-I. Para esta etapa, é proposto o método MCMM (SÁ; NEDJAH; MOURELLE, 2015c). Na ETAPA-IV, é feito o refinamento das posições estimadas, por meio de duas alternativas de algoritmo de otimização por inteligência coletiva: o algoritmo de otimização por enxame de partículas, ou PSO (Particle Swarm Optimization) (ENGELBRECHT, 2006); e o algoritmo de busca por retrocesso, ou BSA (Backtracking Search Optimization Algorithm) (CIVICIOGLU, 2013a). Para compor a função objetivo, minimizada durante a ETAPA-IV, é proposto um novo fator de confiança, denominado MMA-CF (SÁ; NEDJAH; MOURELLE, 2014b), o qual é comparado com o fator de confiança HTA-CF, proposto por (EKBERG, 2009). Tais fatores de confiança visam dar maior importância às contribuições dadas por nós que tendam a ter posições mais precisas. Com base no método proposto, foram desenvolvidos e avaliados quatro algoritmos, resultantes da combinação dos dois algoritmos de otimização, PSO e BSA, com os dois tipos de fator de confiança, MMA-CF e HTA-CF. Os primeiros resultados obtidos durante este trabalho foram publicados em (SÁ et al., 2014; SÁ; NEDJAH; MOURELLE, 2014, 2014a, 2014b, 2015c) e aceitos para publicação em (SÁ; NEDJAH; MOURELLE, 2015a, 2015b).

O restante desta dissertação está organizado em seis capítulos, cujos conteúdos são brevemente descritos a seguir.

Inicialmente, no Capítulo 1, são apresentadas algumas classificações dadas aos métodos de localização alternativos ao GPS, sob diferentes aspectos.

No Capítulo 2 são apresentados diversos trabalhos relacionados ao problema de localização em SRSs, bem como em WSNs. Nesse Capítulo, os trabalhos são organizados segundo as estratégias de uso ou não de técnicas de medição de distância.

O Capítulo 3 discorre sobre os algoritmos de otimização por inteligência coletiva e descreve em detalhes os algoritmos PSO e BSA, utilizados na solução ora proposta para o problema de localização.

No Capítulo 4 é apresentado o método de localização proposto nesta dissertação, o qual é abordado em quatro etapas. São descritos em detalhes os algoritmos Sum-Dist, empregado na ETAPA-I, Min-Max, utilizado na ETAPA-II, e MCMM, proposto para a ETAPA-III. Este capítulo apresenta ainda a modelagem da função objetivo que é minimizada na ETAPA-IV, bem como os dois métodos utilizados para calcular o fator de confiança dos nós da rede.

O Capítulo 5 apresenta os resultados obtidos pelos quatro algoritmos, derivados do método proposto, por meio de simulações e experimentos em robôs reais do tipo *Kilobot*. Os resultados obtidos nas simulações são comparados com os resultados obtidos por outros algoritmos de localização. Tanto nas simulações quanto nos experimentos são explorados problemas com diferentes topologias, quantidades de nós e proporção de nós de referência.

Finalmente, o Capítulo 6 apresenta as conclusões acerca do método proposto e dos quatro algoritmos dele decorrentes, bem como algumas perspectivas e possibilidades de trabalhos futuros.

Capítulo 1

MÉTODOS DE LOCALIZAÇÃO

A necessidade de conhecer a posição dos robôs de um SRS, ou mesmo dos sensores de uma WSN, aliada às restrições de uso de receptores de GPS nestes dispositivos, têm motivado a busca por métodos alternativos para solucionar o problema de localização. Neste capítulo, são apresentadas algumas classificações dadas a estes métodos alternativos de localização, sob diferentes aspectos. Na Seção 1.1, são abordados métodos de localização que independem de medidas de distância. Na Seção 1.2, são considerados os métodos em que a localização é feita com base em medidas de distância, bem como são apresentadas algumas técnicas comumente empregadas na medição de distância. Na Seção 1.3, são feitas considerações quanto a dois tipos de localização: relativa ou absoluta. Na Seção 1.4, são feitas considerações quanto a duas formas de processamento: centralizado ou distribuído. Finalmente, na Seção 1.5, são feitas considerações a respeito do método de localização proposto nesta dissertação.

1.1 Localização Independente de Medidas de Distância

A localização independente de medidas de distância, ou *range-free*, geralmente utiliza informação de conectividade dos nós, ao invés de considerar as medidas de distância entre eles. A conectividade, em conjunto com a informação de posição dos nós de referência, são normalmente utilizadas para estimar a posição dos nós desconhecidos (EKBERG, 2009). Neste tipo de localização, determinados algoritmos são capazes de inferir algum grau de proximidade entre os nós da rede, embora a distância entre eles não possa ser medida diretamente.

Em geral, a localização independente de medidas de distância apresenta como

vantagem a simplicidade e, em alguns casos, a economia de recursos. Porém, seus resultados são menos acurados do que no caso da localização baseada em medidas de distância (EKBERG, 2009). Exemplos deste tipo de localização são apresentados na Seção 2.2.

1.2 Localização Baseada em Medidas de Distância

Muitos algoritmos de localização, possivelmente a maioria (EKBERG, 2009), fazem uso de medidas de distância entre os nós da rede. Para estimar a posição dos nós desconhecidos, tais medidas de distância são normalmente utilizadas em conjunto com a informação de posição dos nós de referência, cujas posições são conhecidas. Alguns exemplos de localização baseada em medidas de distância são descritos na Seção 2.1.

Existem diferentes maneiras de medir a distância entre os elementos de um SRS ou de uma WSN. Nesta seção, apresentamos algumas das técnicas mais comuns utilizadas. Nas Seções 1.2.1, 1.2.2 e 1.2.3, são descritas técnicas de medição de distância que se baseiam no tempo de propagação de sinais. Na Seção 1.2.4, é descrita uma técnica de medição de distância que se baseia nas medidas de potência de sinais recebidos.

1.2.1 Tempo de Propagação do Sinal em uma Direção

Dentre as formas de medição de distância em sistemas sem fio, há um conjunto de técnicas que se baseia na medição do tempo de propagação dos sinais. Uma delas, conceitualmente simples, consiste na medição de distância pelo tempo de propagação do sinal em uma direção. Nesta técnica, o transmissor envia uma mensagem contendo a informação do tempo em que ocorreu a transmissão t_{TX}. Na recepção, o receptor compara o tempo em que a mensagem foi transmitida t_{TX}, com o tempo em que a mesma foi recebida t_{RX} (MAO; FIDAN; ANDERSON, 2007). Com estas informações, o nó receptor é capaz de calcular a distância d para o transmissor por meio da Equação 1:

$$d = s(t_{RX} - t_{TX}), \tag{1}$$

onde s é a velocidade de propagação do sinal no meio.

Esta técnica requer que os relógios do transmissor e do receptor estejam precisamente sincronizados, o que pode onerar o custo do sistema, ou mesmo aumentar a complexidade da rede, pela inclusão de mecanismos de sincronia entre os nós. Tal requisito de sincronismo torna esta técnica uma opção pouco atrativa em WSNs (MAO; FIDAN; ANDERSON, 2007).

1.2.2 Tempo de Propagação do Sinal em duas Direções

A medida de distância pelo tempo de propagação do sinal em duas direções se propõe a solucionar o problema de sincronismo enfrentado quando se mede distâncias pelo tempo de propagação do sinal em apenas uma direção. Nesta técnica, o tempo de propagação compreende o percurso de ida e volta do sinal, sendo computado no mesmo nó que originou a mensagem, com base no seu próprio relógio. Neste caso, a distância entre o nó transmissor e o nó receptor é calculada de acordo com a Equação 2:

$$d = \frac{s(t_{RX} - t_{TX} - t_P)}{2}, \tag{2}$$

onde t_{TX} é o tempo em que a mensagem foi transmitida pelo nó de origem, t_P é o tempo que o outro nó leva para processar a mensagem e retransmitir de volta para o nó de origem e t_{RX} é o tempo em que a mensagem foi recebida de volta no nó de origem. Apesar deste método apresentar uma solução para o problema de sincronismo, existe nele o desafio de conhecer t_P com boa precisão. Para equacionar este problema, normalmente, se calibra t_P com um valor previamente definido, ou o mesmo é medido e reportado em tempo real para que seja computado na Equação 2 (EKBERG, 2009).

1.2.3 Tempo de Propagação de dois Sinais em uma Direção

Nesta técnica, dois sinais com velocidades de propagação diferentes são transmitidos simultaneamente (MAO; FIDAN; ANDERSON, 2007). Devido às diferentes velocidades de propagação, os sinais, que foram transmitidos simultaneamente, serão recebidos em diferentes instantes de tempo. A partir da diferença de tempo de chegada, é possível calcular a distância entre os nós de acordo com a Equação 3 (EKBERG, 2009),

$$d = s_1 s_2 \left(\frac{t_{RX2} - t_{RX1}}{s_1 - s_2} \right), \tag{3}$$

onde s_1 e t_{RX1} são a velocidade de propagação e o tempo de chegada do sinal mais rápido, respectivamente, e s_2 e t_{RX2} são a velocidade de propagação e o tempo de chegada do sinal mais lento, respectivamente.

Um exemplo desta técnica é apresentado em (PRIYANTHA; CHAKRABORTY; BALAKRISHNAN, 2000), onde a distância é medida com base na diferença de tempo de chegada de dois sinais: um transmitido por RF (s_1) e outro transmitido por ultrassom(s_2). Neste caso, como a velocidade de propagação do sinal de RF é muito maior do que a velocidade

de propagação do ultrassom, *i.e.* $s_1 >> s_2$, a Equação 3 pode ser simplificada conforme a Equação 4 (EKBERG, 2009):

$$d = s_2(t_{RX2} - t_{RX1}) \tag{4}$$

Esta técnica apresenta a vantagem de não depender do sincronismo dos nós da rede, conforme apresentado na Seção 1.2.1, tampouco necessita do conhecimento de t_P, conforme apresentado na Seção 1.2.2. Entretanto, o uso de dois sinais que se propaguem em velocidades diferentes requer que cada dispositivo seja dotado de dois sistemas de comunicação, o que reflete em um aumento do custo e do tamanho do dispositivo.

1.2.4 Potência do Sinal Recebido

Uma das técnicas de medição de distância mais utilizada em WSN se baseia na medida de potência dos sinais de RF. Os sensores de uma WSN, em geral, se comunicam por meio de transmissores e receptores de RF, os quais também são capazes de medir a potência do sinal recebido, ou RSS (*Received Signal Strength*) (SICHITIU; RAMADURAI; PEDDABACHAGARI, 2003). A partir do modelo apresentado na Equação 5, o qual é amplamente utilizado para expressar a relação entre a distância e a RSS (LYMBEROPOULOS; LINDSEY; SAVVIDES, 2006), um nó receptor é capaz de estimar a sua distância para o nó transmissor. Dessa forma, a medida de distância pode ser feita durante o processo de comunicação, sem a necessidade de um *hardware* adicional para tal fim, o que torna a técnica atrativa no que concerne ao consumo de potência, custo e tamanho dos nós (LYMBEROPOULOS; LINDSEY; SAVVIDES, 2006):

$$RSS(d) = P_T - \overline{L}(d_0) - 10\psi log_{10}\left(\frac{d}{d_0}\right) + X_\sigma, \tag{5}$$

onde P_T é a potência de transmissão, $\overline{L}(d_0)$ é perda de percurso, conhecida para uma distância de referência d_0 do transmissor, ψ é o expoente de perda de percurso e X_σ é uma variável aleatória de distribuição normal, com média zero e variância σ^2.

Do ponto de vista de custo, os sistemas baseados na RSS de sinais de RF são vantajosos (LANGENDOEN; REIJERS, 2003). Entretanto, nestes sistemas, as medidas de distância não são tão acuradas quanto aquelas baseadas no tempo de propagação do sinal, particularmente quando são aplicados sinais acústicos e sinais de RF combinados (SAVVIDES; PARK; SRIVASTAVA, 2002; GIROD; ESTRIN, 2001), conforme descrito na Seção 1.2.3.

De forma similar aos sistemas que utilizam a RSS de sinais de RF, os robôs de um SRS, eventualmente, utilizam medidas de potência de sinais infravermelhos, ou IR (*Infrared*), para medir distâncias (RUBENSTEIN; AHLER; NAGPAL, 2012; MONDADA et al., 2009; BENET et al., 2002; KORBA; ELGAZZAR; WELCH, 1994; NOVOTNY; FERRIER, 1999). Em alguns casos, os robôs utilizam sensores de IR exclusivamente para a medição de distância (MONDADA et al., 2009; BENET et al., 2002; KORBA; ELGAZZAR; WELCH, 1994; NOVOTNY; FERRIER, 1999), seja para obstáculos ou para outros robôs. Em outros casos, como em (RUBENSTEIN; AHLER; NAGPAL, 2012), os transceptores IR são utilizados tanto para a comunicação quanto para a medição de distância entre robôs.

1.3 Localização Relativa ou Absoluta

Em alguns sistemas de localização, não é necessário que os nós se localizem segundo um sistema de coordenadas específico, desde que os mesmos se localizem segundo um mesmo sistema de coordenadas. Neste tipo de localização, denominada relativa, os nós devem ser capazes de encontrar suas coordenadas dentro de uma topologia de rede, ou enxame, entretanto, tais coordenadas não podem ser correlacionadas com o mundo real. Apesar disto, a localização relativa pode ser suficiente para muitas aplicações, como, por exemplo, no roteamento geográfico de dados em WSN (NICULESCU; NATH, 2003), ou mesmo quando se deseja estabelecer comportamentos de formação em SRS (RUBENSTEIN, 2009).

Em oposição aos sistemas de localização relativa, num sistema de localização absoluta, as coordenadas dos nós são definidas segundo um sistema de coordenadas global previamente definido, como o de latitudes e longitudes, o que permite um mapeamento direto dos nós com o mundo real. Na localização absoluta, a localização dos nós desconhecidos é feita com base na posição de um conjunto de nós de referência, também denominados âncoras, cujas coordenadas são conhecidas e estão referenciadas a tal sistema de coordenadas global. Para isto, os nós de referência podem ser dotados de receptores GPS, ou mesmo serem programados com as coordenadas de onde foram precisamente depositados. Na maioria dos algoritmos de localização, os nós são localizados de forma absoluta (BULUSU; HEIDEMANN; ESTRIN, 2000; NICULESCU; NATH, 2001; RABAEY; LANGENDOEN, 2002; SAVVIDES; PARK; SRIVASTAVA, 2002; EKBERG, 2009).

Há ainda sistemas onde, primeiramente, os nós se localizam de forma relativa para, num segundo passo, se localizarem de forma absoluta, caso haja referências na rede

(SHANG et al., 2003).

1.4 Localização Distribuída ou Centralizada

Os problemas de localização podem ser solucionados por uma estratégia de processamento centralizado (SAVVIDES; PARK; SRIVASTAVA, 2002; SHANG et al., 2003; ZHANG et al., 2008) ou distribuído (NICULESCU; NATH, 2001; SAVVIDES; PARK; SRIVASTAVA, 2002; RABAEY; LANGENDOEN, 2002; EKBERG, 2009; RUBENSTEIN, 2009). Na estratégia de processamento centralizado, todos os dados relevantes da rede, tais como conectividade e distância, são reunidos em uma única estação central para processamento. Esta estação central é responsável por calcular a posição e retornar os resultados a todos os nós da rede. A vantagem da estratégia de processamento centralizado é que é possível reduzir os requisitos computacionais dos nós da rede, sejam eles sensores de uma WSN ou robôs de um SRS. Entretanto, a estação central deve possuir mais recursos computacionais, para que possa suportar a carga de processamento de localização a ela delegada. Há de se considerar que, no caso de falha na estação central, todos os nós da rede ficam impossibilitados de conhecer a sua localização, o que representa uma vulnerabilidade do sistema de localização centralizada.

Na estratégia de processamento distribuído, não há uma estação central para calcular a localização de todos os nós da rede. Neste caso, o processamento é feito de forma distribuída por todos os nós. Normalmente, cada nó deve ser capaz de estimar sua própria posição e ajudar os seus vizinhos a calcular as suas respectivas posições. Os sistemas de localização distribuídos apresentam maior escalabilidade, uma vez que, ao aumentar o tamanho da rede, não é necessário aumentar os requisitos de *hardware* de uma estação central de processamento. Na estratégia de processamento centralizado, o custo computacional demandado na estação central aumenta com o tamanho da rede, ao passo que, no processamento distribuído, o custo computacional em cada nó da rede se mantém constante, independente do tamanho da rede, considerando, em ambos os casos, redes que apresentem a mesma conectividade. Outro ponto favorável dos sistemas de localização distribuídos é a sua robustez, uma vez que uma eventual falha em um nó na rede, na maioria dos casos, não impede que os demais nós da rede estimem sua localização.

1.5 Considerações Finais

Neste capítulo foram abordadas algumas classificações e características dos métodos de localização normalmente adotados como alternativa ao uso do GPS. No que se refere aos conceitos apresentados, podemos classificar o método proposto nesta dissertação como sendo de localização absoluta, baseada em medidas de distância, cuja solução é obtida através de um processamento distribuído. O método de localização proposto é descrito em detalhes no Capítulo 5.

Capítulo 2

TRABALHOS RELACIONADOS

ESTE capítulo apresenta alguns trabalhos relacionados ao problema de localização, desenvolvidos como alternativas ao GPS, para ambientes internos ou sistemas cujos elementos possuam limitações que impeçam o uso de tal sistema de posicionamento. As soluções são organizadas nas Seções 2.1 e 2.2, segundo duas possíveis estratégias: a de localização baseada em medidas de distância e a de localização independente de medidas de distância, respectivamente. Na Seção 2.3, são apresentadas algumas considerações finais.

2.1 Localização Baseada em Medidas de Distância

Os métodos de localização baseados em medidas de distância, ou *range-based*, utilizam tais medidas para inferir a localização dos nós desconhecidos. Conforme apresentado em (MAO; FIDAN; ANDERSON, 2007), estas informações podem ser obtidas por meio de medidas de RSS, TOA (*Time of Arrival*) e TDOA (*Time Difference of Arrival*). Os métodos *range-based*, em geral, proporcionam maior acurácia que os métodos *range-free* (YUN et al., 2009), entretanto, requerem que os nós sejam dotados de *hardware* capaz de efetuar as medidas de distância.

Em (LANGENDOEN; REIJERS, 2003), é feita a análise de três algoritmos de localização distribuídos, que usam métodos *range-based* : *Ad-hoc positioning* (NICULESCU; NATH, 2001), *Robust Positioning* (RABAEY; LANGENDOEN, 2002) e *N-hop Multilateration* (SAVVIDES; PARK; SRIVASTAVA, 2002). Os autores identificam uma estrutura comum aos três algoritmos e, com base nesta constatação, propõem uma estrutura genérica para algoritmos de localização distribuídos, a qual é dividida em três etapas:

- ETAPA-I: Estimar as distâncias de cada nó desconhecido para cada nó de referência.

- ETAPA-II: Calcular a posição inicial de cada nó desconhecido, a partir das distâncias estimadas na ETAPA-I.
- ETAPA-III: Refinar a posição de cada nó desconhecido, a partir das informações de posição e distância dos nós vizinhos.

Nas Seções 2.1.1, 2.1.2, 2.1.3, 2.1.4 e 2.1.5, são apresentados alguns métodos de localização à luz das três etapas supramencionadas. Na Seção 2.1.6, é apresentado um método de localização estruturado de forma diferente das referidas etapas.

2.1.1 Método de Posicionamento *Ad-hoc*

Em (NICULESCU; NATH, 2001), os autores propõem o algoritmo *Ad-hoc positioning*, no qual três métodos são avaliados para uso na primeira etapa: DV-Hop, DV-Distance e Euclideano. No método DV-HOP, também denominado Hop-TERRAIN (RABAEY; LANGENDOEN, 2002), o nó desconhecido calcula a distância para um nó de referência multiplicando o número de saltos entre ambos pelo comprimento médio de um salto. O comprimento médio de um salto é obtido pela razão da distância entre os nós de referência e o número de saltos medidos entre estes nós. A distância entre os nós de referência é calculada em função de suas coordenadas conhecidas. O método DV-Distance é semelhante ao DV-Hop, com a diferença de que a distância entre os nós é medida diretamente em metros ao invés de saltos. Neste método, a distância para um nó de referência é obtida pelo somatório do comprimento de cada salto entre o nó desconhecido e o nó de referência. O terceiro método, Euclideano, se baseia na geometria local dos nós. Quando um nó desconhecido conhece a distância de dois vizinhos para um nó de referência, bem como as distâncias entre estes vizinhos, o nó desconhecido é capaz de calcular sua distância para o nó de referência. Comparado com os métodos baseados em vetor de distância, DV-Hop e DV-Distance, o método Euclideano proporciona uma melhor acurácia para topologias onde a conectividade não é isotrópica e, geralmente, possui desempenho mais previsível. Porém, o método Euclideano possui um maior custo de comunicação.

Na segunda etapa do algoritmo *Ad-hoc positioning*, a posição de cada nó é calculada de forma analítica, por meio do método da *Lateração*. Este método estabelece um sistema de equações lineares que pode ser solucionado por meio do método dos mínimos quadrados, conforme a Equação 6:

$$x = (A^T A)^{-1} A^T b, \tag{6}$$

onde x representa as coordenadas do nó desconhecido, A é uma matriz calculada em função das coordenadas dos nós de referência e b é uma matriz calculada em função das coordenadas e das distâncias medidas para os nós de referência. Entretanto, há situações excepcionais em que o método falha, em virtude de $(A^T A)$ ser não inversível. A terceira etapa, de refinamento, não é executada pelo algoritmo *Ad-hoc positioning*.

2.1.2 Método de Posicionamento Robusto

O algoritmo *Robust Positioning*, proposto em (RABAEY; LANGENDOEN, 2002), utiliza os métodos DV-Hop e *Lateração* na sua primeira e segunda etapa, respectivamente. Adicionalmente, este algoritmo executa a terceira etapa, de refinamento. O refinamento proposto pelos autores consiste em um processo iterativo onde, a cada ciclo, cada nó desconhecido recalcula, pelo método da *Lateração*, a sua posição em relação à sua vizinhança. Cabe ressaltar que nesta etapa, a *Lateração* é feita com base nas posições e distâncias de vizinhos que estão a um salto de distância, sejam eles nós desconhecidos ou de referência, diferentemente do que é feito na segunda etapa, onde a *Lateração* é feita apenas com as posições e distâncias dos nós de referência.

2.1.3 Método de Localização em Múltiplos Saltos com Multilateração

O algoritmo *N-hop Multilateration* (SAVVIDES; PARK; SRIVASTAVA, 2002) utiliza, em sua primeira etapa, o método Sum-Dist para calcular a distância para os nós de referência. Assim como o DV-Distance, este método realiza a soma do comprimento de cada salto percorrido por uma mensagem originada nos nós de referência. Assume-se, então, que a distância do nó de referência é igual ao comprimento do caminho mais curto percorrido por tal mensagem. Na segunda etapa, o *N-hop Multilateration* utiliza o método Min-Max, que apresenta como vantagem a simplicidade e o menor número de operações com ponto flutuante, quando comparado ao método da *Lateração*. A idéia principal deste método é criar regiões cujas fronteiras são calculadas com base na posição e na distância estimada de cada nó de referência, e, em seguida, determinar a interseção dessas regiões. A posição do nó desconhecido, nesta etapa, é definida como sendo o ponto central da região de interseção. Na terceira etapa, o algoritmo faz o refinamento da posição por meio do método da *Lateração*, utilizando as posições e distâncias obtidas de vizinhos que estão a um salto de distância, de forma similar ao que é feito no algoritmo *Robust Positioning*.

O algoritmo desenvolvido nesta dissertação também foi dividido em etapas, seguindo uma abordagem similar àquela dada por (LANGENDOEN; REIJERS, 2003). No projeto do algoritmo, levou-se em conta o fato de que o mesmo seria implementado em robôs do tipo *Kilobot*, que possuem limitações de *hardware* e de fonte de energia. Tais limitações foram determinantes para a escolha dos métodos empregados nas duas primeiras etapas do algoritmo. Na primeira etapa, optou-se pelo método Sum-Dist, por ser o método que demanda menor custo computacional e de comunicação (LANGENDOEN; REIJERS, 2003), quando comparado aos métodos DV-Hop e Euclideano. Na segunda etapa, optou-se pelo método Min-Max por ser mais simples e demandar menor número de operações com ponto flutuante (LANGENDOEN; REIJERS, 2003), quando comparado ao método da *Lateração*.

2.1.4 Localização por Inteligência de Enxame

A estrutura genérica de três etapas também pode ser identificada em outros trabalhos não contemplados no estudo feito por (LANGENDOEN; REIJERS, 2003). Um exemplo é o algoritmo de Localização por Inteligência de Enxame, ou SIL (*Swarm-Intelligent Localization*), proposto por (EKBERG, 2009) para solucionar problemas de localização em WSN com múltiplos saltos, utilizando o PSO. O SIL também pode ser analisado sob o prisma das três etapas de (LANGENDOEN; REIJERS, 2003), apesar de, originalmente, não ter sido descrito desta forma. Na primeira etapa, é utilizado o método Sum-Dist para determinar as distâncias para os nós de referência. Na segunda etapa, cada nó desconhecido estima sua posição inicial por meio do PSO, o qual minimiza uma função objetivo que leva em conta as informações de posição e distância dos nós de referência. Tal função objetivo é modelada com base nos preceitos da Geometria Euclidiana de Distância ou EDG (LIBERTI et al., 2014) (*Euclidean Distance Geometry*). Na terceira etapa, de refinamento, o PSO é utilizado para minimizar uma função objetivo, também modelada com base na EDG, que leva em consideração informações locais de vizinhos a um e dois saltos de distância, bem como informações dos nós de referência. Após minimizar a função objetivo, os nós trocam mensagens contendo suas novas posições encontradas. As novas posições informadas pelos nós vizinhos são utilizadas para atualizar a função objetivo. Uma vez atualizada a função objetivo, a mesma é novamente minimizada pelo PSO. Este processo se repete até que o critério de parada seja atendido. Tendo em vista a baixa complexidade do PSO e

das funções objetivo, o algoritmo apresenta as vantagens de poder ser escrito em poucas linhas de código e de apresentar operações de baixa complexidade, sendo a raiz quadrada a operação mais complexa utilizada.

O SIL foi testado por meio de simulações, apresentadas em (EKBERG, 2009), nas quais foram considerados erros de medição de distância. Os resultados mostram que os erros de localização crescem mais lentamente do que os erros de medição, o que indica que o algoritmo é relativamente robusto, em face de erros de medição de distância. No que tange à conectividade da rede, os resultados mostram que quanto maior a conectividade, menores são os erros de localização. Da mesma forma, o aumento do numero de nós de referência conduz a menores erros de localização, de maneira que resultados aceitáveis são obtidos com uma proporção de nós de referência de aproximadamente 10% da rede.

O uso de um algoritmo de otimização, como o PSO, na terceira etapa do processo de localização, apresenta vantagens quando comparado ao método da *Lateração*. Com um algoritmo de otimização é possível, por meio da modelagem da função objetivo, agregar mais informações relevantes ao cálculo da localização, como, por exemplo, informações de vizinhos a dois saltos de distância. Por este motivo, nesta dissertação, também optou-se pelo uso de um algoritmo de otimização na terceira etapa do algoritmo de localização. Não obstante, com o objetivo de melhorar o desempenho da terceira etapa, são avaliados nesta dissertação outros algoritmos de otimização, bem como são propostos aprimoramentos na função objetivo adotada em (EKBERG, 2009).

2.1.5 Método Baseado em Gauss-Newton

Outro método é apresentado em (CHENG; VANDENBERGHE; YAO, 2009). Neste trabalho, o autor propõe um algoritmo de localização baseado no algoritmo de Gauss-Newton (DENNIS; SCHNABEL, 1996). O algoritmo de Gauss-Newton é um algoritmo iterativo usado para resolver problemas de mínimos quadrados não lineares, podendo ser visto como uma modificação do método de Newton para achar o mínimo de uma função. O algoritmo de localização proposto assume que as distâncias para os nós de referência já tenham sido estimadas por métodos como DV-Hop, DV-Distance ou Sum-dist. Da mesma forma, considera-se que as posições iniciais dos nós desconhecidos já tenham sido estimadas por métodos como Min-Max ou *Lateração*. A contribuição do trabalho está na proposição de um método alternativo para execução da terceira etapa de (LANGENDOEN; REIJERS,

2003), ou seja, a etapa de refinamento. Deste modo, o algoritmo de Gauss-Newton é utilizado para encontrar o mínimo de uma função objetivo que leva em consideração informações de posição e distância de nós vizinhos a um salto de distância, bem como dos nós de referência. Após minimizar a função objetivo, os nós trocam mensagens contendo suas novas posições encontradas. As novas posições informadas pelos nós vizinhos são utilizadas para atualizar a função objetivo. Uma vez atualizada a função objetivo, a mesma é novamente minimizada pelo algoritmo de Gauss-Newton. Este processo se repete até que a atualização da posição seja menor que um limiar pré-definido.

2.1.6 Método Baseado na Trilateração Distribuída

Uma outra solução para o problema de localização é apresentada em (RUBENSTEIN, 2009). Em seu trabalho, o autor desenvolve um sistema de localização *range-based*, com processamento distribuído, com o objetivo de permitir que um enxame de robôs possa estabelecer comportamentos de auto-formação e de recomposição. Neste caso, um sistema de coordenadas local, *i.e.* relativo aos outros robôs do enxame, é suficiente, não sendo necessário que a localização seja absoluta. Diferente dos demais trabalhos abordados na Seção 2.1, seu método não é estruturado nas três etapas de (LANGENDOEN; REIJERS, 2003). Sua primeira etapa consiste em estabelecer um sistema de coordenadas local. Para isto, o autor utiliza um método denominado *trilateration seeding*, o qual estabelece um sistema de coordenadas a partir da posição relativa de três robôs não colineares. Primeiramente, um robô é escolhido para ser a origem, com coordenadas $(0, 0)$. Em seguida, um robô vizinho é escolhido para formar o eixo x do sistema de coordenadas. A coordenada do segundo robô é $(d, 0)$, onde d é a distância medida para o robô que está na origem. Após isto, um terceiro robô, que possui contato com os dois primeiros, passa a ter duas possíveis coordenadas. O terceiro robô escolhe, então, a coordenada que possui o valor de y positivo. Com a definição da posição do terceiro robô, o sistema de coordenadas local é estabelecido. Uma vez estabelecido o sistema de coordenadas local, qualquer robô que possa medir sua distância aos três primeiros pode facilmente estimar sua posição pelo método da Trilateração. Em virtude do processamento distribuído, é natural que mais de um robô inicie o processo de *trilateration seeding*. Isto resulta em múltiplos sistemas de coordenadas que se tornam conflitantes à medida que se propagam pelo enxame por meio da trilateração. Para resolver tais conflitos, o autor apresenta um método, denomi-

nado *coordinate merging*, capaz de criar um sistema único de coordenadas independente de quantos robôs tenham iniciado o processo de *trilateration seeding*. Para estabelecer um sistema único, cada sistema de coordenadas deve ser transladado para o mesmo ponto e rotacionado para a mesma orientação. No que diz respeito a translação, cada sistema de coordenadas é transladado para o ponto médio de todos os sistemas de coordenadas utilizando um vetor de translação $\vec{V}_{seed}$. Da mesma forma, no que diz respeito à rotação, cada sistema de coordenadas é rotacionado para a orientação média de todos os sistemas de coordenadas por meio de uma matriz de rotação R_{seed}. Finalmente, conhecendo $\vec{V}_{seed}$ e R_{seed}, cada robô do enxame converte suas coordenadas para o novo sistema de coordenadas, agora único. Uma desvantagem do método proposto por (RUBENSTEIN, 2009) é que cada robô necessita, obrigatoriamente, de ao menos três robôs com posição conhecida em sua vizinhança, para estabelecer sua posição por meio da Trilateração.

Uma vantagem do método proposto nesta dissertação, quando comparado com o desenvolvido por (RUBENSTEIN, 2009), é que um nó desconhecido não necessita, obrigatoriamente, ter três vizinhos com posição conhecida para calcular sua posição. Neste caso, a ausência de um terceiro nó adjacente é, em alguns casos, compensada pela adição de outras contribuições na função objetivo, como, por exemplo, a de nós que estão a dois saltos de distância.

2.2 Localização Independente de Medidas de Distância

Conforme descrito na Seção 1.1, a localização independente de medidas de distância, ou *range-free*, é usada para encontrar a localização dos nós desconhecidos sem a necessidade de obter informações de distância entre os nós da rede. Nas seções 2.2.1, 2.2.2, 2.2.3, 2.2.4 e 2.2.5, são apresentado alguns métodos de localização *range-free*.

2.2.1 Método por RFID

Em (TESORIERO et al., 2010), é apresentada uma solução *range-free* para rastreamento de robôs, ou outras entidades, em ambientes internos, utilizando a tecnologia de RFID (*Radio Frequency Identification*). No sistema proposto, a superfície em que o robô se move é dividida em pequenos quadrados, aqui denominados células. O tamanho da célula define a resolução do sistema. Um número único de identificação é atribuído a cada célula por

meio de uma etiqueta RFID passiva. No robô, é instalado um leitor de RFID, que faz a leitura do número da etiqueta RFID da célula em que o mesmo se encontra. Este número é transmitido por comunicação sem fio a um gerenciador de localização, que é responsável por mapear a posição do robô em função do número informado.

Esta solução tem como vantagem a grande simplicidade, não sendo necessário nenhum cálculo para estabelecer a posição do robô. Entretanto, o sistema tem como desvantagem a falta de autonomia do robô para identificar sua posição, sendo necessário um elemento central para processar e mapear as posições dos robôs. Outra limitação é a necessidade de uma matriz de etiquetas RFID passivas em todo o ambiente de operação, o que restringe o uso dos robôs a ambientes em que haja tal infraestrutura. Além disso, a quantidade de etiquetas RFID aumenta com a resolução da localização. Na solução proposta nesta dissertação, os robôs são capazes de estabelecer suas posições sem a necessidade de haver uma infraestrutura especial no ambiente. Não são necessárias referências externas ao enxame, nem um elemento central para localizar os robôs.

2.2.2 Localização por Média Aritmética

Em (BULUSU; HEIDEMANN; ESTRIN, 2000), os autores propõem outro método *range-free* para estimar a localização de nós desconhecidos que estão a um salto de distância dos nós de referência. Neste método, os nós de referência são dispostos em uma malha regular e suas regiões de cobertura se sobrepõem. Cada nó de referência transmite, periodicamente, mensagens contendo suas coordenadas. Ao receber tais mensagens, cada nó desconhecido computa sua posição como sendo o centróide das posições de todos os nós de referência que estão ao seu alcance. Ou seja, para cada dimensão, a coordenada do nó desconhecido é definida pela média aritmética das coordenadas dos nós de referência que estão ao seu alcance. Este método é simples e demanda poucos recursos, porém apresenta grande quantidade de erro de posição (YUN et al., 2009).

2.2.3 Localização por Média Ponderada

Em (KIM; KWON, 2005), os autores propõem uma versão aprimorada do método proposto por (BULUSU; HEIDEMANN; ESTRIN, 2000). Da mesma forma que naquele método, os nós de referência divulgam suas coordenadas para os nós desconhecidos que estão a um salto de distância. Entretanto, neste método, a posição dos nós desconhecidos não é

obtida calculando o centróide, *i.e.* a média aritmética, da posição dos nós de referência adjacentes. Neste método, os autores introduzem um peso que é dado aos nós de referência de acordo com sua proximidade aos nós desconhecidos. Assim, para cada dimensão, as coordenadas do nó desconhecido são calculadas pela média ponderada das coordenadas dos nós de referência a que estão conectados. O sucesso deste método depende da escolha apropriada dos pesos dados aos nós de referência, o que é feito de forma heurística.

2.2.4 Localização por Métodos de Inteligência Computacional

Em (YUN et al., 2009), os autores propõem duas soluções *range-free* para o problema de localização em WSN. A primeira solução utiliza como base o método desenvolvido por (KIM; KWON, 2005), onde a posição de cada nó desconhecido é calculada a partir da média ponderada da posição dos nós de referência adjacentes. Neste caso, o valor dos pesos de cada nó de referência é calculado por um sistema de lógica *fuzzy* ou FLS (*Fuzzy Logic System*), que tem como entrada a potência do sinal recebido ou RSS dos nós de referência. Cabe ressaltar que o dado de entrada do FLS é simplesmente a potência do sinal recebido, e não uma medida de distância. A faixa de valores de entrada do FLS, compreendida entre $[0, RSS_{max}]$, onde RSS_{max} é o valor máximo da RSS, é decomposta em cinco funções de pertinência trapezoidais. A variável de saída do FLS é o peso atribuído ao nó de referência. Assim como no sinal de entrada, a saída é dividida em cinco funções de pertinência. A lógica *fuzzy* do sistema estabelece uma correlação direta entre cada um dos cinco níveis de RSS da entrada, com cada um dos cinco valores de peso na saída do FLS. Dessa forma, um sinal com baixa RSS na entrada resultará num baixo valor de peso na saída. Um sinal com alta RSS na entrada resultará num alto valor de peso na saída. Para otimizar o FLS, os autores utilizam Algoritmos Genéticos, ou GA (*Genetic Algorithm*). Por simplicidade, apenas a função de pertinência da RSS de entrada é otimizada pelo GA. A lógica do sistema e as funções de saída não são otimizadas.

A segunda solução *range-free* de (YUN et al., 2009) trata de um sistema onde a posição do nó desconhecido é obtida por meio de uma rede neural que tem como entrada a RSS dos nós de referência adjacentes, *i.e.* a um salto de distância. Os nós de referência da rede são dispostos em uma malha regular, quadrada, onde as suas regiões de cobertura se sobrepõem. Esta malha deve ser capaz de cobrir toda a rede de sensores. Cada nó desconhecido é dotado de uma rede neural, com quatro entradas e uma saída. As entradas

são as RSSs dos nós de referência adjacentes que, pela topologia da rede e alcance dos nós, serão, no máximo, quatro. A saída é a posição relativa do nó desconhecido em relação a uma das referências. A rede neural é treinada por meio de *back-propagation*. Para treinar a rede neural, quatro nós de referência são dispostos na forma de um quadrado, e o nó desconhecido, que está sendo treinado, é alocado em diferentes regiões dentro deste quadrado. Desta forma, o treinamento reproduz as condições que o nó encontrará quando estiver em operação. Após ter sua rede neural treinada, o nó é capaz de estimar sua posição baseando-se apenas na RSS dos quatro nós de referência mais próximos.

Em (YUN et al., 2009), as duas soluções, uma usando FLS/GA e a outra redes neurais, são avaliadas por meio de simulações e experimentos reais. As simulações foram conduzidas no MATLAB utilizando redes com 120 nós de referência e 60 nós desconhecidos, numa área de 100 $\times$ 100 m^2. Nas simulações, o erro médio de posicionamento foi de 0,7119 m, na solução por redes neurais, e 0,7802 m, na solução por FLS/GA. Nos experimentos, o erro médio de posicionamento foi de 0,6938 m, na solução por redes neurais, e 0,8025 m, na solução por FLS/GA. Os resultados mostram que a solução por redes neurais é mais acurada que a solução por FLS/GA, sendo que ambas apresentam desempenho superior às soluções propostas por (BULUSU; HEIDEMANN; ESTRIN, 2000) e (KIM; KWON, 2005). Entretanto, o elevado número de nós de referência, que nas simulações foi de 66,7% da rede, representa uma desvantagem para ambos os métodos.

2.2.5 Localização Baseada na Assinatura de RSSs

Um método de localização *range-free* que contorna o problema da grande quantidade de nós de referência é apresentado em (STELLA; RUSSO; BEGUŠIĆ, 2014). Os autores apresentam um algoritmo de localização de dispositivos em ambientes internos, em duas dimensões, baseado na análise da assinatura, ou *fingerprint*, do RSS proveniente de um conjunto de nós de referência. Neste trabalho, os nós de referência são dispositivos sem fio heterogêneos de tecnologias como o GSM (*Global System for Mobile Communications*), UMTS (*Universal Mobile Telecommunication System*) e WLAN (*Wireless Local Area Network*). A técnica proposta presume que cada posição possui uma assinatura única de RSS, $a = [f_1, f_2, f_3, ...f_k] \in R^k$, onde f é a medida do RSS de um nó de referência e k é a quantidade de nós de referência. O processo é composto de duas fases: a fase de coleta de dados; e a fase de localização em tempo real. A primeira fase consiste em

medir a RSS em toda a área de operação, mapeando a assinatura do RSS em função da localização. Ao fim desta fase, haverá um mapa de RSS, em duas dimensões, para cada nó de referência. Na segunda fase, o nó desconhecido mede a assinatura de RSS do local em que se encontra. O nó desconhecido calcula então, para cada nó de referência, um mapa de erro de RSS, que corresponde ao módulo da diferença entre o mapa de RSS e a RSS medida. Desse modo, no mapa de erro de RSS, as regiões com erro zero são regiões onde possivelmente o nó desconhecido está. Fazendo a média entre os mapas de erro de RSS de cada nó de referência, obtêm-se um novo mapa, onde o ponto de valor mínimo corresponde à posição em que o nó desconhecido está. Como o método proposto visa a aplicação em redes heterogêneas, é necessário atribuir um peso à contribuição de cada nó de referência em função da sua tecnologia. Dessa forma, os autores propõem uma média ponderada dos mapas de erro de RSS, onde os pesos são calculados em função do inverso da variância das medidas de RSS de cada tecnologia.

O método proposto por (STELLA; RUSSO; BEGUŠIĆ, 2014) apresenta como vantagem o uso de nós de referência pertencentes a uma infra-estrutura de rede comum, normalmente já disponível nos locais de operação, como nos casos das tecnologias GSM, UMTS e WLAN. Adicionalmente, o algoritmo apresenta baixa complexidade computacional. Entretanto, o método apresenta como desvantagem a necessidade de um mapeamento prévio da RSS em todo o local de aplicação. No método proposto nesta dissertação, as referências são robôs que conhecem suas posições e que fazem parte do enxame, o que torna a solução independente de uma infra-estrutura local. Neste caso, não há a necessidade do uso de uma referência externa, como redes GSM, UMTS ou WLAN, nem há a necessidade do mapeamento da assinatura do RSS no local de operação.

2.3 Considerações Finais

Neste capítulo, foram apresentados trabalhos relacionados ao problema de localização que servem como alternativa ao uso do GPS, principalmente para aplicações em ambientes internos ou que envolvam robôs ou sensores incapazes de comportar um receptor GPS.

Os métodos foram classificados segundo duas possíveis estratégias: *range-based* e *range-free*. Os métodos *range-based* estudados apresentam a vantagem de não necessitar de nenhuma infra-estrutura externa ao enxame para estabelecer a localização dos robôs, tampouco necessitam de um levantamento da assinatura de RSS no local de operação.

Isto torna este tipo de solução mais flexível no que diz respeito ao ambiente de operação. Outra vantagem é o fato do erro ser, em geral, menor em redes do tipo *range-based* do que em redes do tipo *range-free*. Em contrapartida, é possível constatar que os algoritmos de localização *range-based*, em geral, apresentam um maior custo computacional, quando comparados aos métodos *range-free*.

As soluções *range-free*, como as apresentadas em (BULUSU; HEIDEMANN; ESTRIN, 2000), (KIM; KWON, 2005) e (YUN et al., 2009), em geral, demandam mais nós de referência que as soluções *range-based* apresentadas na Seção 2.1. Isto se deve ao fato de que, nas soluções *range-free* apresentadas, os nós de referência precisam estar a um salto de distância de cada nó desconhecido da rede para que este possa estabelecer sua posição. Sendo assim, é possível constatar que, nestas soluções, a quantidade de nós de referência é proporcional à área da rede e inversamente proporcional ao raio de alcance dos nós de referência. A título de exemplo, nas simulações com soluções *range-free*, apresentadas em (YUN et al., 2009), cerca de 66,7% dos nós da rede são nós de referência, enquanto que em (EKBERG, 2009) o autor indica que resultados aceitáveis foram obtidos com 10% de nós de referência na rede.

Nesta dissertação, optou-se pelo uso de uma solução *range-based*, tendo em vista que este tipo de solução, em geral, apresenta resultados mais acurados que as soluções *range-free* (YUN et al., 2009) e demanda uma menor proporção de nós de referência, quando comparada com os métodos apresentados em (BULUSU; HEIDEMANN; ESTRIN, 2000), (KIM; KWON, 2005) e (YUN et al., 2009).

As vantagens apresentadas pelos métodos *range-based* foram determinantes na escolha por este tipo de método, uma vez que o óbice relacionado ao seu custo computacional não é um fator impeditivo para a implementação em robôs do tipo *Kilobot*, empregados nos experimentos desta dissertação. Adicionalmente, os robôs *Kilobot* são dotados de sensores capazes de medir a distância entre robôs vizinhos, o que favorece aos métodos *range-based*.

Capítulo 3

ALGORITMOS DE OTIMIZAÇÃO POR INTELIGÊNCIA COLETIVA

OS algoritmos de otimização por inteligência coletiva consistem em métodos estocásticos capazes de resolver problemas não-lineares, não-diferenciáveis e problemas de otimização numérica complexos (CIVICIOGLU, 2013a), que, muitas vezes, carecem de uma solução analítica, ou mesmo apresentam soluções analíticas de alto custo computacional. Uma solução alternativa para estes tipos de problema reside na busca exaustiva, onde as chances de se aproximar da solução ótima depende exclusivamente da granularidade das amostras do espaço de busca. A desvantagem da busca exaustiva é seu elevado custo computacional, requerido quando a granularidade se torna mais fina. Diferente da busca exaustiva, os algoritmos de inteligência coletiva são capazes de desempenhar uma busca adaptativa pela solução ótima, utilizando o conhecimento adquirido por um conjunto de agentes que navegam pelo espaço de busca. A busca adaptativa permite a redução do número de avaliações feitas no espaço de busca, resultando num menor custo computacional, quando comparada à busca exaustiva. De uma forma geral os algoritmos de inteligência coletiva são capazes de alcançar resultados ótimos, ou próximos do ótimo.

3.1 Visão Geral

Os algoritmos de inteligência coletiva tiveram origem no estudo de colônias ou enxame de organismos sociais como, por exemplo, abelhas, formigas, peixes e aves. De uma forma geral, estes sistemas naturais evidenciam a capacidade de um grupo social encontrar soluções para um problema que estaria além da capacidade de um único indivíduo. Nestes sistemas, indivíduos simples, dotados de pouca inteligência e recursos sensoriais muitas vezes limitados, compartilham as informações adquiridas com outros indivíduos os quais

reajustam seus comportamentos para que o coletivo convirja para uma solução ótima desejada.

As formigas, por exemplo, são capazes de encontrar o caminho mais curto para o alimento com base em um conjunto de regras comportamentais simples e um sistema de comunicação onde a troca de informações é feita pelo depósito de feromônio no meio (ENGELBRECHT, 2007). O comportamento das formigas em busca de alimento inspirou o desenvolvimento do algoritmo de otimização por colônia de formigas, ou ACO (*Ant Colony Optimization*) (DORIGO; BIRATTARI, 2010), utilizado para otimizar problemas combinatórios como, por exemplo, o problema do caixeiro viajante (BONABEAU; DORIGO; THERAULAZ, 1999).

Outro bem sucedido sistema baseado na inteligência coletiva é o praticado pelas abelhas na busca e extração de néctar. Neste sistema, abelhas forrageiras (que procuram por comida), ao encontrarem uma boa área para extração de néctar, retornam para a colmeia onde realizam uma dança denominada *waggle dance*. Por meio desta dança, as forrageiras compartilham informações sobre a direção, distância e quantidade de néctar de uma determinada fonte, recrutando outras abelhas para exploração da área descoberta. O comportamento coletivo das abelhas também foi traduzido em um conjunto de regras que deu origem ao algoritmo de colônias de abelhas artificiais, ou ABC (*Artificial Bee Colony Algorithm*) (CHAN; TIWARI, 2007), proposto para solucionar problemas de otimização.

A graciosa e imprevisível coreografia de um conjunto de aves capazes de mudar repentinamente de direção, voltando a se agrupar em uma formação ótima, inspirou o desenvolvimento do PSO (KENNEDY J. E EBERHART, 1995; ENGELBRECHT, 2006). Inicialmente, o enxame de partículas foi desenvolvido para simular graficamente o comportamento social de pássaros voando em bando, emulando os padrões que governam tal voo síncrono. A partir deste objetivo inicial, o modelo evoluiu para o PSO, um algoritmo de otimização simples e eficiente.

Mais recentemente, em (CIVICIOGLU, 2013a), foi proposto um novo algoritmo de inteligência coletiva denominado algoritmo de busca por retrocesso, ou BSA. Tal algoritmo, por sua vez, encontra inspiração no comportamento de grupos sociais de criaturas que retornam, em intervalos aleatórios, às regiões previamente conhecidas como boas para obtenção de alimento.

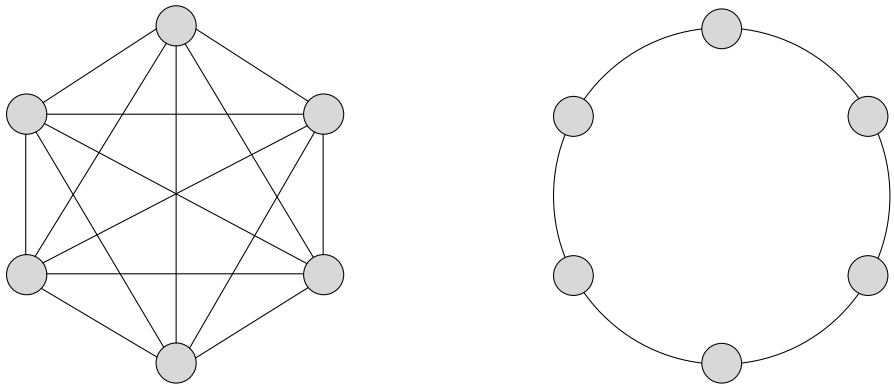
Nesta dissertação, são empregados dois algoritmos de inteligência coletiva, o PSO

e o BSA, os quais são descritos em detalhes nas Seções 3.2 e 3.3, respectivamente.

3.2 Otimização por Enxame de Partículas

O PSO tem origem na simulação de modelos sociais simplificados, como, por exemplo, a movimentação de cardumes de peixes ou bandos de aves (KENNEDY J. E EBERHART, 1995; SHI; EBERHART, 1998), sendo o termo partícula utilizado para generalizar o indivíduo. Nesta analogia, uma partícula representa uma possível solução para um problema, em um domínio multidimensional. O enxame, por sua vez, representa um conjunto de possíveis soluções. Cada partícula possui coordenadas, que definem a sua posição no espaço de busca, bem como uma aptidão, que corresponde ao valor da função objetivo naquela posição. A função objetivo expressa o problema que se pretende minimizar ou maximizar.

Na busca adaptativa pela melhor solução, cada partícula, ao projetar o seu deslocamento, utiliza tanto a própria experiência como a experiência compartilhada por partículas vizinhas. Inicialmente, duas versões do PSO foram desenvolvidas, o *Gbest* e o *Lbest* PSO, os quais se diferem pela definição da vizinhança da partícula (ENGELBRECHT, 2006). No *Gbest* PSO, a vizinhança de cada partícula é definida como sendo todo o enxame, refletindo uma topologia de rede social em estrela, conforme a Figura 1a. Já o *Lbest* PSO utiliza uma topologia de rede social em anel, conforme a Figura 1b, resultando em uma menor vizinhança, quando comparado ao *Gbest* PSO. A topologia de rede social adotada nesta dissertação, no caso do PSO, é a topologia em estrela, pelo fato desta apresentar uma maior velocidade de convergência que a topologia em anel (ENGELBRECHT, 2007).



(a) *Estrela* (b) *Anel*

Figura 1: Topologias de redes sociais

Para navegar no espaço de busca, a cada ciclo iterativo, cada partícula atualiza

sua posição conforme a Equação 7, onde χ_i e ν_i são a posição e a velocidade da partícula i, respectivamente:

$$\chi_i(t+1) = \chi_i(t) + \nu_i(t+1). \tag{7}$$

O vetor de velocidade ν_i de cada partícula é calculado pela soma de três componentes: uma componente cognitiva, uma componente social e uma componente de inércia do movimento. A componente cognitiva considera a melhor posição encontrada pela partícula ao longo de sua trajetória, constituindo, assim, a memória da partícula. Esta componente adiciona à trajetória da partícula uma tendência de retorno à melhor posição por ela visitada. A componente social decorre da troca das experiências adquiridas entre as partículas do enxame. Dessa forma, numa rede social com topologia em estrela, essa componente visa conduzir a partícula à melhor solução global, *i.e.* a melhor solução encontrada pelo enxame até o momento. A componente de inércia, por sua vez, corresponde a uma parcela do deslocamento anterior que é preservada para o deslocamento subsequente, de acordo com um peso denominado *coeficiente de inércia*, introduzido por (SHI; EBERHART, 1998). Sendo assim, o cálculo da velocidade da uma partícula é feito conforme a Equação 8:

$$\nu_{ij}(t+1) = \omega\nu_{ij}(t) + \varphi_1 r_{1j}(t)(m_{ij} - \chi_{ij}(t)) + \varphi_2 r_{2j}(t)(m_{gj} - \chi_{ij}(t)), \tag{8}$$

onde ν_{ij} é a velocidade da partícula i em cada dimensão j, ω é o coeficiente de inércia, φ_1 e φ_2 são os coeficientes que atribuem pesos às componentes cognitiva e social, respectivamente, r_{1j} e r_{2j} são números aleatórios entre [0,1], m_{ij} é a melhor posição da partícula i na dimensão j e m_{gj} é a melhor posição global do enxame na dimensão j. A Figura 2 ilustra geometricamente o cálculo da velocidade de uma partícula.

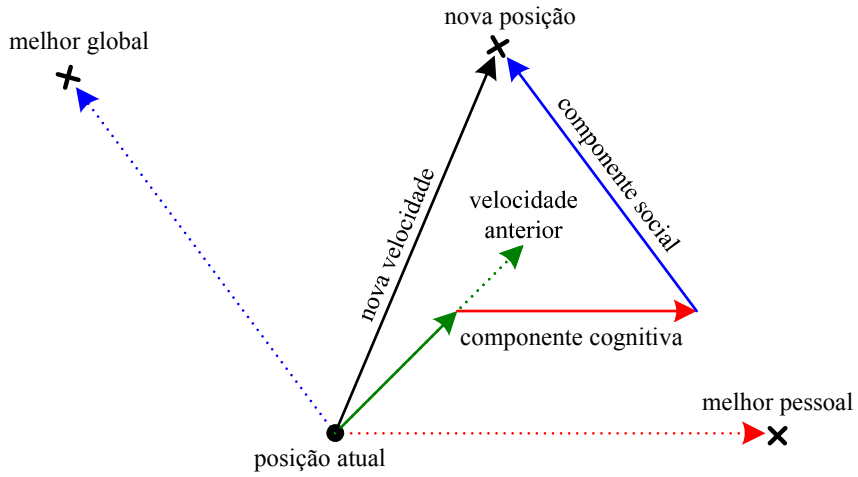


Figura 2: Cálculo da velocidade de uma partícula do PSO

Um aspecto importante que determina a eficiência de um algoritmo de otimização é a sua capacidade de exploração e de refinamento. A capacidade de exploração corresponde à habilidade do algoritmo em procurar soluções em diferentes regiões do espaço de busca, reduzindo a probabilidade de convergência para um mínimo local. Por outro lado, a capacidade de refinamento resulta da habilidade do algoritmo em concentrar a busca em torno de uma região promissora, aumentando a probabilidade de se aproximar ou, até mesmo, encontrar o valor ótimo. Ocorre que, nas primeiras aplicações do PSO, verificou-se que a velocidade calculada por meio da Equação 8 tendia a crescer rapidamente, especialmente para partículas afastadas das melhores posições global e individual (ENGELBRECHT, 2006). Velocidades altas fazem com que as partículas transponham, a cada iteração, grande parcela do espaço de busca sem devidamente sondá-lo, reduzindo a capacidade exploratória do algoritmo. Sendo assim, para controlar a capacidade exploratória do algoritmo, é estabelecido um limite de velocidade (EBERHART; SIMPSON; DOBBINS, 1996; ENGELBRECHT, 2006) para cada dimensão j, dado pela Equação 9:

$$0 \leq \nu_{ij} \leq \delta(max_j - min_j), \tag{9}$$

onde max_j e min_j são os limites máximo e mínimo do espaço de busca na dimensão j, e $\delta \in [0, 1]$ é uma constante ajustada empiricamente.

A estrutura geral do PSO é definida pelo Algoritmo 1. O primeiro passo do algoritmo é a inicialização do enxame e a avaliação da aptidão de cada partícula, estabelecendo os valores iniciais de m_i e m_g. Normalmente, na inicialização, as partículas são espalhadas aleatoriamente por todo o espaço de busca, utilizando uma distribuição uniforme (ENGELBRECHT, 2006). Em seguida, ocorre o processo iterativo do algoritmo. Neste processo, a cada iteração, a posição de cada partícula é atualizada, de acordo com as Equações 7, 8 e 9, e a aptidão da mesma é reavaliada com base nas coordenadas da nova posição visitada. Assim, se a aptidão da partícula for a melhor aptidão obtida por ela até o momento, a posição visitada passa a ser considerada a melhor posição da partícula m_i. Se a aptidão da partícula for a melhor aptidão obtida pelo enxame, até o momento, a posição visitada passa a ser considerada a melhor posição global m_g. O processo iterativo do algoritmo se repete até que a condição de parada seja satisfeita. Usualmente, é adotada uma das seguintes condições de parada (ENGELBRECHT, 2006):

- *Número máximo de iterações.* Este critério é útil quando se deseja avaliar a melhor solução que o algoritmo é capaz de encontrar dentro de um determinado tempo. É

notório que esta condição de parada pode findar o processo de otimização antes que uma boa solução seja encontrada, caso o número de iterações seja pequeno. Por outro lado, um grande número de iterações pode elevar desnecessariamente o custo computacional, considerando situações em que uma boa solução pode ser encontrada com poucas iterações.

- *Obtenção de uma solução aceitável.* Neste critério, a otimização termina quando a diferença entre a solução ótima e a melhor solução encontrada é menor do que um limiar de erro predefinido. Para isto, esta condição de parada requer que o valor ótimo seja previamente conhecido, o que não ocorre para muitas aplicações.

- *Nenhuma melhoria é observada durante um determinado número de iterações.* Existem diferentes formas de verificar a estagnação do processo de otimização. Por exemplo, se a velocidade média das partículas for pequena, pode-se considerar que houve a convergência do enxame. Outra forma consiste em verificar a ausência de melhora significativa na aptidão da melhor solução, durante um determinado número de iterações.

Algoritmo 1 PSO

```
início
    Inicia o enxame com χ e ν aleatórias;
    Avalia o enxame;
    Armazena m_i e m_g;
    repita
        Atualiza velocidades, conforme Equações 8 e 9;
        Atualiza posições, conforme Equação 7;
        Avalia o enxame;
        Atualiza m_i e m_g, caso melhores posições tenham sido encontradas;
    até Condição de parada;
    retorna m_g;
fim
```

3.3 Algoritmo de Busca por Retrocesso

O BSA é um novo algoritmo evolucionário (CIVICIOGLU, 2013a), que utiliza informações obtidas por gerações passadas para buscar soluções de melhor aptidão. A filosofia bio-inspirada do BSA é análoga a um grupo social de seres vivos que, em intervalos aleatórios, retornam às zonas de caça que foram previamente consideradas boas para a obtenção

de alimentos. A estrutura geral do BSA, composta por cinco etapas, é apresentada no Algoritmo 2.

Algoritmo 2 BSA

```
início
    Inicialização;
    repita
        Seleção-I;
        Gera nova população
            Mutação;
            Cruzamento;
        fim
        Seleção-II;
    até Condição de parada;
fim
```

Na etapa Inicialização, o algoritmo gera e avalia a população inicial P_0, e inicia uma população histórica P_{hist}. A população histórica constitui a memória do BSA. Nesta etapa, os indivíduos de ambas as populações P_0 e P_{hist} são gerados de forma aleatória, utilizando uma distribuição uniforme para todo o espaço de busca.

Durante a Seleção-I, o algoritmo determina, aleatoriamente, se a população atual P será gravada como população histórica P_{hist}. A decisão de armazenar a população atual como a população histórica, ou memória do enxame, é feita conforme a Equação 10:

$$P_{hist} = \begin{cases} P & \text{Se } (a < b); \\ \\ P_{hist} & \text{Se } (a \geq b), \end{cases} \tag{10}$$

onde $a, b \sim U(0,1)$, sendo U uma distribuição uniforme. Cabe ressaltar que a Equação 10 garante que a população memorizada será lembrada até que uma nova população seja armazenada. Ainda nesta etapa, os indivíduos de P_{hist} são embaralhados, independente do resultado da Equação 10.

No BSA, o deslocamento dos indivíduos pelo espaço de busca se dá pelas etapas Mutação e Cruzamento. O produto final destas duas operações é uma nova população, denominada P_{nova}. Na etapa Mutação, é criada uma população modificada P_{mod} (versão inicial de P_{nova}), de acordo com a Equação 11. Dessa forma, P_{mod} é o resultado do movimento dos indivíduos de P nas direções estabelecidas por $(P_{hist} - P)$. S representa a amplitude do deslocamento, que, neste algoritmo, é dada pela Equação 12:

$$P_{mod} = P + S(P_{hist} - P) \tag{11}$$

$$S = K \cdot \Gamma \tag{12}$$

O valor de K é ajustado empiricamente via uma série de simulações, já $\Gamma \sim \mathrm{N}(0, 1)$, onde N é a distribuição normal padrão. Note que, de acordo com a Equação 11, cada indivíduo do BSA compartilha sua lembrança com apenas um outro indivíduo. A cada iteração, cada indivíduo projeta seu deslocamento com base no conhecimento de um novo par, uma vez que P_{hist}, que representa memória do enxame, é sempre embaralhada na etapa Seleção-I. A Figura 3 representa graficamente o cálculo de P_{mod} em uma população com seis indivíduos.

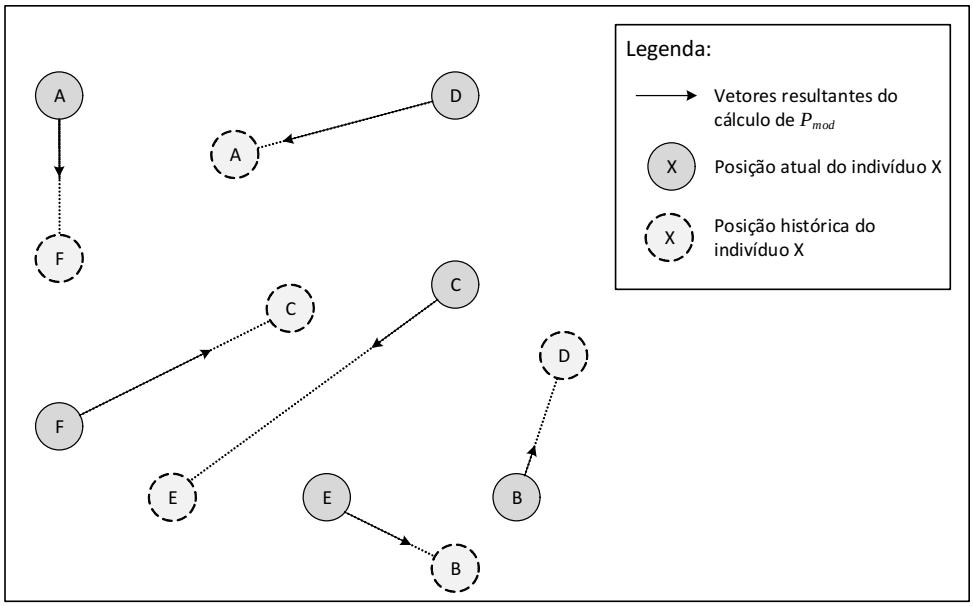


Figura 3: Representação gráfica do cálculo de P_{mod}

A etapa Cruzamento é responsável por gerar a versão final da nova população P_{nova}. Nesta etapa, os elementos de P_{mod} são combinados com os elementos de P, de acordo com duas estratégias escolhidas aleatoriamente. Em ambas as estratégias, cada indivíduo de P_{nova} tem parte de suas coordenadas oriundas de P_{mod} e as demais oriundas de P. Na primeira estratégia, a escolha das dimensões que receberão coordenadas de P_{mod} é feita de forma aleatória, podendo ser mais de uma dimensão. Na segunda estratégia, cada indivíduo de P_{nova} receberá apenas uma de suas coordenadas provenientes de P_{mod}, as demais coordenadas serão as originais, oriundas de P. A operação de cruzamento é descrita pelo Algoritmo 3 (CIVICIOGLU, 2013a), onde X é a matriz de cruzamento, Y é o número de indivíduos de P, D é o número de dimensões do problema, $rnd \sim U(0, 1)$,

$rndi(D) \sim U(0, D)$, sendo $rndi(D) \in \mathbb{Z}$, e Q é um fator de cruzamento que define o número máximo de dimensões do indivíduo de P_{nova} que receberá coordenadas de P_{mod}.

Algoritmo 3 Operação de cruzamento do BSA

início
  $X_{(1:Y,1:D)} = 1$;
  se $a < b \mid a, b \sim U(0, 1)$ **então**
    Estratégia-I:
    para $i = 1 : Y$ **faça**
      $X_{i,u_{(1:[Q \cdot rnd \cdot D])}} = 0 \mid u =$*permutação*$(\langle 1, 2, 3, ..., D \rangle)$;
    fim
  fim
  senão
    Estratégia-II:
    para $i = 1 : Y$ **faça**
      $X_{i,rndi(D)} = 0$;
    fim
  fim
  $P_{nova} = P_{mod}$
  para $i = 1 : Y$ **faça**
    para $j = 1 : D$ **faça**
      se $X_{i,j} = 1$ **então**
        $P_{nova_{i,j}} = P_{i,j}$
      fim
    fim
  fim
fim

Na etapa Cruzamento, é feito ainda um controle para que os indivíduos de P_{nova} não excedam as fronteiras do espaço de busca. Caso algum indivíduo exceda os limites de uma dada dimensão do espaço de busca, sua coordenada é realocada aleatoriamente, utilizando uma distribuição uniforme, dentro dos limites da respectiva dimensão.

Na etapa Seleção-II, o algoritmo seleciona os elementos de P_{nova} (indivíduos obtidos após mutação e cruzamento) que tenham a aptidão melhor do que os de P (indivíduos antes da mutação e do cruzamento) e os substitui em P. Dessa forma, P só incluirá novos indivíduos que tenham evoluído, *i.e.* indivíduos que tenham encontrado locais com maior oferta de alimentos.

As etapas Seleção-I, Mutação, Cruzamento e Seleção-II se repetem iterativamente, até que a condição de parada seja satisfeita. Em sua proposição original (CIVICIOGLU, 2013a), o BSA emprega como condição de parada um número máximo de iterações. Entretanto, cabe ressaltar a possibilidade de uso de outras condições de parada, tais como

as descritas na Seção 3.2. Uma vez satisfeita a condição de parada, o algoritmo retorna a melhor solução encontrada.

3.4 Considerações Finais

Os algoritmos PSO e BSA, descritos nas Seções 3.2 e 3.3, respectivamente, foram escolhidos para desempenhar a etapa de refinamento do problema de localização, objeto desta dissertação. Tal escolha foi motivada pela confiabilidade e eficiência que ambos têm apresentado na solução de um vasto número de problemas de otimização numérica (ENGELBRECHT, 2006; POLI, 2008; CIVICIOGLU, 2013a), inclusive em problemas de localização (EKBERG, 2009; SÁ et al., 2014; SÁ; NEDJAH; MOURELLE, 2014, 2014a, 2014b, 2015b, 2015a).

Outrossim, a simplicidade de ambos os algoritmos, que podem ser escritos em poucas linhas de código, favorece a implementação em sistemas com reduzidos recursos computacionais, tais como os robôs do tipo *Kilobot* utilizados nos experimentos desta dissertação. O desempenho de ambos os algoritmos é analisado no Capítulo 5, tanto no que se refere à exatidão das soluções do problema de localização, quanto no que diz respeito ao tempo de processamento consumido pelos mesmos.

Capítulo 4

SOLUÇÃO PROPOSTA PARA O PROBLEMA DE LOCALIZAÇÃO

NESTE capítulo, é proposta uma solução para o problema de localização em múltiplos saltos. A solução é apresentada em duas versões, que diferem no algoritmo de otimização por inteligência coletiva empregado para fazer o refinamento das posições. Uma versão utiliza o PSO (SÁ; NEDJAH; MOURELLE, 2014a) e a outra o BSA (SÁ; NEDJAH; MOURELLE, 2014b). Para compor a função objetivo, minimizada pelos referidos algoritmos de otimização, é introduzida uma nova técnica para estimar a confiança atribuída às informações de localização recebidas da vizinhança (SÁ; NEDJAH; MOURELLE, 2014b). A solução desenvolvida reúne ainda os métodos Sum-Dist e Min-Max, tradicionalmente empregados nos problemas de localização, bem como um novo método denominado Localização Colaborativa Min-Max em Múltiplos Saltos, ou MCMM (SÁ; NEDJAH; MOURELLE, 2015c).

Primeiramente, na Seção 4.1, é apresentada a descrição do problema de localização tratado nesta dissertação. Posteriormente, na Seção 4.2, é proposta uma solução para o referido problema, tendo como premissa as condições estabelecidas na Seção 4.1.

4.1 Descrição do Problema

O problema de localização em tela consiste em inferir a posição de um conjunto de nós desconhecidos I a partir de informações de posição de um conjunto de nós referência R e de medidas de distância entre os nós da rede. Os nós desconhecidos representam robôs ou sensores cujas posições não são conhecidas. Já os nós de referência são nós cujas posições são previamente conhecidas, seja pelo fato de estes nós dotarem de receptores GPS, ou sistema similar, ou por terem sido precisamente depositados em posições predefinidas. O

fato de os nós de referência possuírem coordenadas conhecidas, que podem ser referenciadas a um sistema de coordenadas global, faz com que o problema seja de localização absoluta, conforme definido na Seção 1.3.

As técnicas de medição de distância comumente empregadas se baseiam na RSS, no tempo de propagação de um sinal ou no tempo de propagação de dois sinais com diferentes velocidades de propagação (MAO; FIDAN; ANDERSON, 2007; LYMBEROPOULOS; LINDSEY; SAVVIDES, 2006). Uma vez que as referidas técnicas de medição de distância dependem de características de propagação de sinal, há de se considerar uma distância limite para tais medidas. No caso mais simples, onde todos os nós de referência estão dentro da distância limite de medição, as distâncias para os nós de referência são medidas diretamente em um único salto. Entretanto, nos casos onde um ou mais nós de referência estão além da distância limite de medição, as distâncias para estes nós são estimadas indiretamente em múltiplos saltos.

Para que um nó desconhecido seja capaz de estimar sua posição, considerando apenas as referências que estão a um salto de distância, é necessário que a condição estabelecida pela Equação 13 (EKBERG, 2009) seja verdadeira:

$$\rho \geq D + 1, \tag{13}$$

onde ρ é o número de referências que estão a um salto de distância do nó desconhecido e D é o número de dimensões do problema. Se a condição estabelecida pela Equação 13 não for atendida, o problema admite soluções ambíguas, sendo apenas uma verdadeira. A Figura 4 ilustra, a título de exemplo, um caso de ambiguidade quando $\rho = 2$ e $D = 2$. Sendo assim, a necessidade de haver pelo menos $D + 1$ nós de referência a um salto de distância de cada nó desconhecido, majora a demanda por tais dispositivos em problemas cujas medidas são tomadas em um único salto.

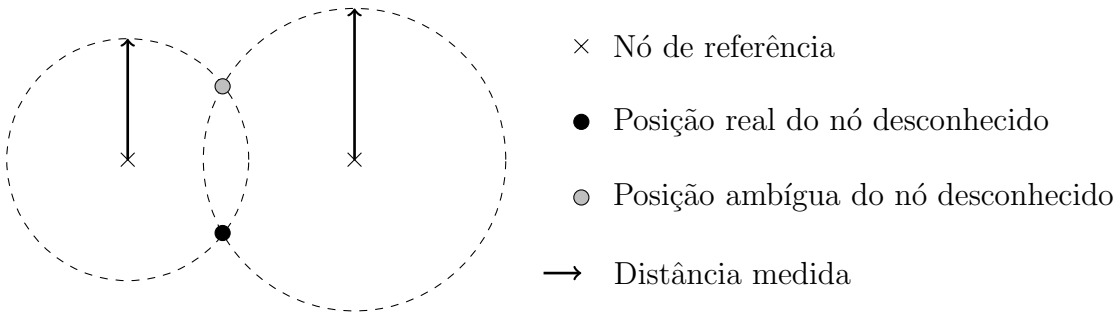


Figura 4: Exemplo de ambiguidade quando $\rho = 2$ e $D = 2$

O problema de localização, abordado nesta dissertação, se refere ao caso em que a localização é feita em múltiplos saltos. Cabe ressaltar que, neste caso, também são admitidas situações em que há nós de referência a apenas um salto de distância. Uma vantagem da localização em múltiplos saltos, conforme a ser apresentado na Seção 4.2, é que a mesma não requer a existência de $D + 1$ nós de referência a um salto de distância de cada nó desconhecido, reduzindo a demanda por nós de referência. O desafio deste problema reside no fato de que a distância estimada em múltiplos saltos para os nós de referência, ocasionalmente, não corresponde à distância real, em virtude do eventual desalinhamento dos nós que formam o caminho para as referências. Isto fica evidente no exemplo da Figura 5, onde $d_{i,3} \neq d_{i,1} + d_{1,2} + d_{2,3}$.

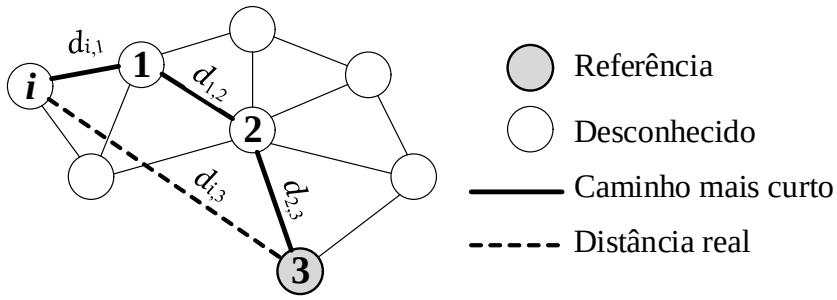


Figura 5: Erro de estimativa de distância por desalinhamento dos nós

Por outro lado, se os dispositivos próximos a um nó desconhecido i forem capazes de estimar suas próprias posições, ainda que com baixa acurácia, o nó i poderá inferir sua posição com base em medidas de distância diretas, feitas para a sua vizinhança. Neste caso, o erro da posição inferida pelo nó i dependerá apenas da acurácia das posições de sua vizinhança, não sendo impactada por possíveis erros de medida causados por caminhos desalinhados. Isto conduz, portanto, a um problema de processamento distribuído, onde cada nó deve contribuir iterativamente com sua vizinhança para que haja uma redução dos erros de posição do sistema.

Em face ao exposto, o problema tratado nesta dissertação é classificado como de localização absoluta, baseado em medidas de distância, de múltiplos saltos e distribuído. O espaço de busca do referido problema está contido em duas dimensões. Cabe ressaltar que, apesar de a solução a ser proposta na Seção 4.2 ser aplicável a problemas com duas ou três dimensões, adotou-se o caso de duas dimensões apenas para simplificar a análise, não prejudicando a generalidade.

4.2 Solução Proposta

A solução ora proposta tem como base a abordagem de três etapas introduzidas por (LANGENDOEN; REIJERS, 2003) e apresentada na Seção 2.1. Entretanto, uma etapa de pré-refinamento foi adicionada ao processo, o qual passa a ser estruturado em quatro etapas:

- ETAPA-I: Estimar as distâncias de cada nó desconhecido para os nós de referência. Nesta etapa é utilizado o método Sum-Dist.
- ETAPA-II: Calcular a posição inicial de cada nó desconhecido a partir das distâncias estimadas na ETAPA-I. Nesta etapa é adotado o método Min-Max.
- ETAPA-III: Fazer o pré-refinamento da posição dos nós desconhecidos, por meio do método MCMM, a fim de mitigar o impacto de possíveis falhas da ETAPA-I.
- ETAPA-IV: Refinar a posição de cada nó desconhecido a partir das informações de posição e distância dos nós vizinhos. Nesta etapa são implementados dois algoritmos de otimização, o PSO e o BSA.

Os métodos aplicados nas etapas I, II e III são descritos nas Seções 4.2.1, 4.2.2 e 4.2.3, respectivamente. Na Seção 4.2.4, é apresentada a função objetivo, minimizada pelo PSO e pelo BSA, bem como a forma como estes algoritmos são aplicados ao problema em questão. Na Seção 4.2.5, são apresentadas as duas técnicas utilizadas para estabelecer o fator de confiança atribuído às informações compartilhadas pelos nós vizinhos. Tais fatores de confiança são aplicados na composição da função objetivo. Na Seção 4.2.6, são apresentadas as duas versões completas do algoritmo: uma baseada no PSO e a outra no BSA.

4.2.1 ETAPA-I: Distância para os Nós de Referência

Para estimar a distância para os nós de referência, foi empregado o método Sum-Dist. Originalmente proposto por (SAVVIDES; PARK; SRIVASTAVA, 2002), este método permaneceu sem nome, até ser nomeado Sum-Dist por (LANGENDOEN; REIJERS, 2003).

No início do processo, por um certo período de tempo, todos os nós de referência da rede enviam mensagens contendo a sua identificação r, suas coordenadas $(\hat{x}_r, \hat{y}_r)$

e a distância total percorrida pela mensagem (inicialmente igual a 0). A mensagem é repassada de nó para nó, de forma que a distância medida em cada salto é adicionada à distância total percorrida pela mensagem. A mensagem recebida por um nó só é armazenada/retransmitida caso a distância percorrida pela mesma seja a menor distância conhecida para um dado nó de referência, até o momento. O conjunto R_i, composto pelos nós de referência descobertos pelo nó i, dependerá do tempo de execução desta etapa. Dessa forma, após um certo tempo, cada nó desconhecido $i \in I$ terá conhecimento da posição $(\hat{x}_r, \hat{y}_r)$ de cada nó de referência $r \in R_i$ e da distância da menor rota $l_{i,r}$ existente entre ambos. O processo desempenhado por cada nó desconhecido i, durante a execução do Sum-Dist, é representado pelo Algoritmo 4, onde v é um vizinho a um salto de distância de i, seja $v \in I$ ou $v \in R$, $l_{v,r}$ é a distância estimada entre o nó vizinho v e o nó de referência r, $d_{i,v}$ é a distância medida entre o nó desconhecido i e o nó vizinho v, e T_S é o tempo de execução do Sum-Dist. Cabe ressaltar que a simplicidade do método resulta em um baixo custo computacional e de comunicação (LANGENDOEN; REIJERS, 2003), quando comparado a outros métodos como o *DV-Hop* e o Euclideano (NICULESCU; NATH, 2001).

Algoritmo 4 Sum-Dist no nó desconhecido i

```
início
    enquanto t < T_S faça
        se recebida mensagem de v contendo (r, x̂_r, ŷ_r, l_{v,r}) então
            Mede d_{i,v};
            se r não for conhecido por i então
                l_{i,r} = d_{i,v} + l_{v,r};
                Armazena (r, x̂_r, ŷ_r, l_{i,r});
                Transmite (r, x̂_r, ŷ_r, l_{i,r}) para a vizinhança;
            fim
            se (r for conhecido por i) e (d_{i,v} + l_{v,r} < l_{i,r}) então
                l_{i,r} = d_{i,v} + l_{v,r};
                Atualiza (l_{i,r});
                Transmite (r, x̂_r, ŷ_r, l_{i,r}) para a vizinhança;
            fim
        fim
    fim
fim
```

A Figura 6 apresenta um exemplo em que um nó desconhecido i estima a distância para três nós de referência por meio do método Sum-Dist. Note que $l_{i,r}$, ou seja, a menor distância percorrida por uma mensagem, entre um nó desconhecido i e um nó de referência r, não é necessariamente a distância real entre i e r. Em um cenário com nós distribuídos

aleatoriamente, como o da Figura 6, é comum que os nós que formam o caminho entre i e r não estejam alinhados, fazendo com que $l_{i,r}$ seja maior do que a distância real entre os dois nós.

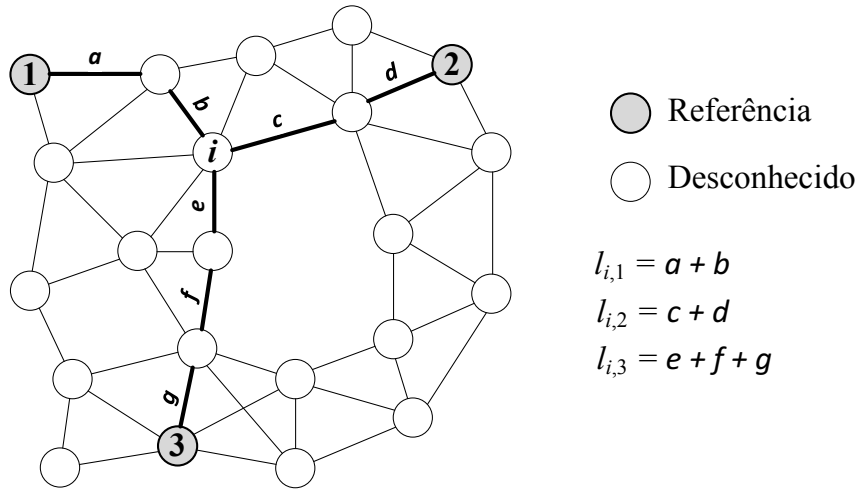


Figura 6: Exemplo de estimativa de distâncias pelo método Sum-Dist

Pelo exposto, qualquer estimativa de posição que seja feita com base nas distâncias $l_{i,r}$, medidas pelo método Sum-Dist, são intrinsecamente de baixa acurácia e precisam ser refinadas. Entretanto, as medidas de distância feitas pelo método Sum-Dist podem subsidiar o cálculo de uma primeira estimativa de posição do nó, ainda que com baixa acurácia, a um baixo custo computacional e de comunicação.

4.2.2 ETAPA-II: Estimativa Inicial da Posição

Nesta etapa, a posição inicial de cada nó desconhecido é calculada por meio do método Min-Max. Este método, originalmente proposto por (SAVVIDES; PARK; SRIVASTAVA, 2002), permaneceu sem nome até receber o nome Min-Max em (LANGENDOEN; REIJERS, 2003).

Neste método, as medidas de distância $l_{i,r}$, obtidas na ETAPA-I, são utilizadas para encontrar os limites de uma região S_i em que o nó i se encontra. A Figura 7 apresenta a forma como os limites dessa região são calculados, considerando somente a dimensão x. Na referida figura, é possível constatar que a distância estimada entre o nó i e o nó de referência r_1 é $l_{i,1} = d_{1,2} + d_{2,i}$, já a distância estimada entre i e o nó de referência r_3 é $l_{i,3} = d_{i,3}$. Sendo $\hat{x}_1$ a coordenada de r_1 na dimensão x, podemos afirmar que, com relação a este nó de referência, a região possível para o nó i está entre $\hat{x}_1 - (d_{1,2} + d_{2,i})$

e $\hat{x}_1 + (d_{1,2} + d_{2,i})$. Fazendo a mesma análise em relação ao nó r_3, podemos afirmar que a região possível para o nó i está compreendida entre $\hat{x}_3 - d_{i,3}$ e $\hat{x}_3 + d_{i,3}$. Considerando ambas as referências simultaneamente, concluímos que a posição do nó i está contida entre $\max[\hat{x}_1 - (d_{1,2} + d_{2,i}), \hat{x}_3 - d_{i,3}] = \hat{x}_3 - d_{i,3}$ e $\min[\hat{x}_1 + (d_{1,2} + d_{2,i}), \hat{x}_3 + d_{i,3}] = \hat{x}_1 + (d_{1,2} + d_{2,i})$. Ou seja, a coordenada estimada x_i do nó i é tal que $\hat{x}_3 - l_{i,3} \leq x_i \leq \hat{x}_1 + l_{i,1}$. O mesmo procedimento se aplica na dimensão y.

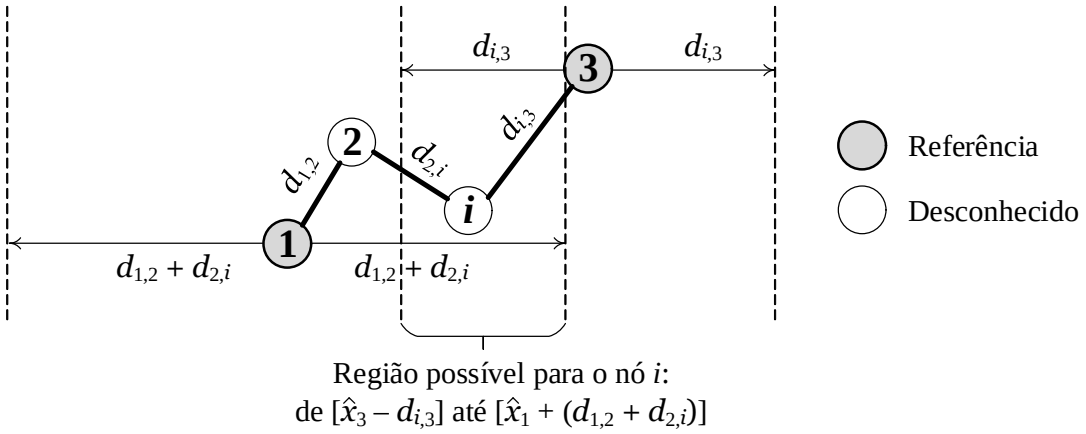


Figura 7: Exemplo do cálculo dos limites de S_i na dimensão x

Com base no exposto, o primeiro passo deste método consiste em criar, para cada par formado por um nó desconhecido i e um nó de referência $r \in R_i$, uma região $B_{i,r}$ denominada *bounding box*. As fronteiras dessa região são calculadas, em cada dimensão, somando e subtraindo $l_{i,r}$ da posição do respectivo nó de referência $(\hat{x}_r, \hat{y}_r)$, conforme a Equação 14:

$$B_{i,r} : [\hat{x}_r - l_{i,r}, \hat{y}_r - l_{i,r}] \times [\hat{x}_r + l_{i,r}, \hat{y}_r + l_{i,r}]. \tag{14}$$

Feito isso, define-se uma área S_i que corresponde à interseção destas regiões, conforme a Equação 15:

$$S_i = \bigcap_{\forall r \in R_i} B_{i,r} : [\max_{\forall r \in R_i}(\hat{x}_r - l_{i,r}), \max_{\forall r \in R_i}(\hat{y}_r - l_{i,r})] \times [\min_{\forall r \in R_i}(\hat{x}_r + l_{i,r}), \min_{\forall r \in R_i}(\hat{y}_r + l_{i,r})]. \tag{15}$$

Finalmente, para determinar a posição inicial u_i, do nó desconhecido i, calcula-se o ponto central de S_i, conforme a Equação 16:

$$u_i = \left(\frac{\max_{\forall r \in R_i}(\hat{x}_r - l_{i,r}) + \min_{\forall r \in R_i}(\hat{x}_r + l_{i,r})}{2}, \frac{\max_{\forall r \in R_i}(\hat{y}_r - l_{i,r}) + \min_{\forall r \in R_i}(\hat{y}_r + l_{i,r})}{2} \right). \tag{16}$$

A Figura 8 apresenta uma representação gráfica do método Min-Max para a estimativa da posição de um nó desconhecido i a partir de três nós de referência r_1, r_2 e r_3.

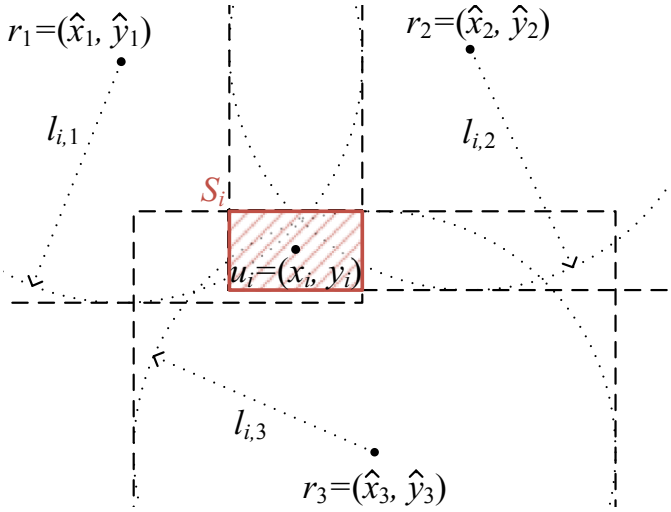
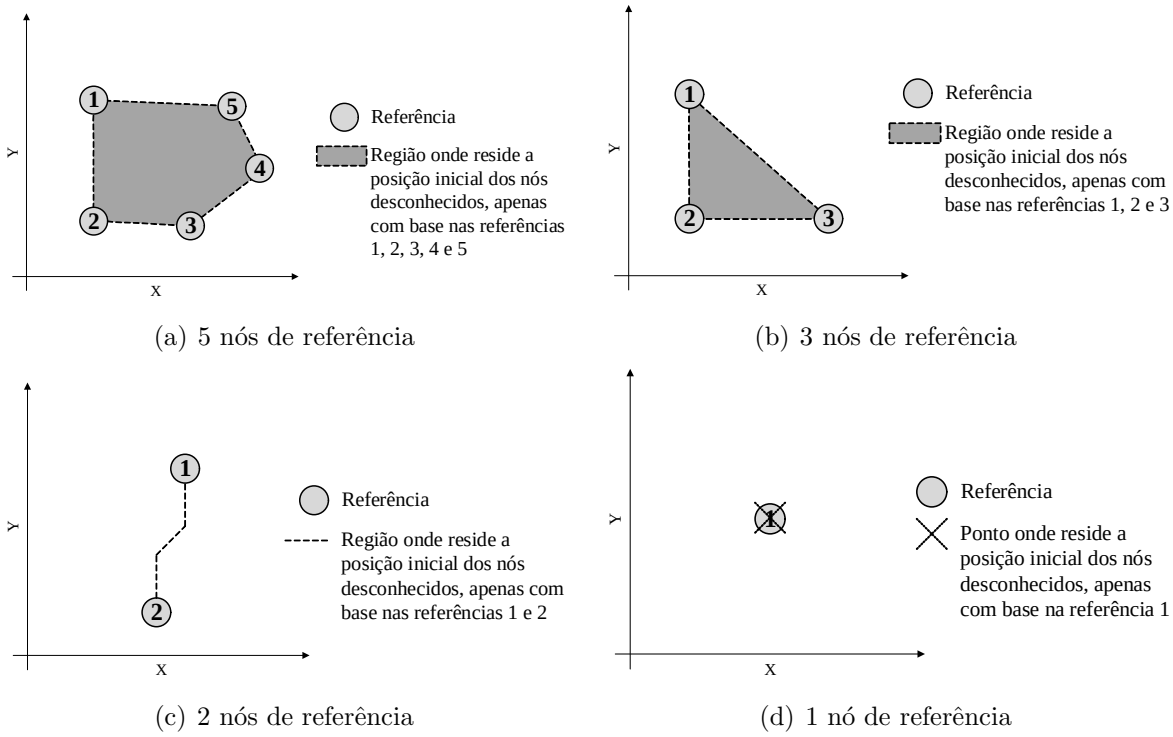


Figura 8: Representação gráfica do método Min-Max

A qualidade da posição inicial encontrada pelo método Min-Max depende da quantidade de nós de referência r descobertos pelo nó desconhecido i, da acurácia das distâncias $l_{i,r}$ estimadas em múltiplos saltos na ETAPA-I e da posição do nó i em relação aos nós de referência $r \in R_i$. Segundo (SAVVIDES; PARK; SRIVASTAVA, 2002), uma limitação do método Min-Max decorre do fato de que as posições por ele estimadas sempre residirão em uma região convexa formada pelos nós de referência considerados no cálculo. Desta forma, se um nó desconhecido i estiver fora da região convexa formada pelos nós de referência por ele conhecidos, *i.e.* $r \in R_i$, a qualidade da posição estimada será comprometida. As Figuras 9a e 9b apresentam exemplos de regiões convexas formadas por cinco e três nós de referência, respectivamente. As regiões demarcadas representam a área aproximada onde residirá a posição estimada por um nó desconhecido i, esteja ele dentro ou fora da referida região. Nos casos em que o nó desconhecido conhece apenas duas referências, a posição estimada pelo Min-Max restará sobre a linha que liga as duas referências, conforme representado na Figura 9c. Neste caso, podemos observar que a linha que liga as duas referências não constitui uma região convexa, sendo este caso uma exceção à regra apresentada por (SAVVIDES; PARK; SRIVASTAVA, 2002). Num caso extremo, em que o nó desconhecido conheça apenas uma referência, a posição estimada para este nó terá as mesmas coordenadas do nó de referência conhecido, conforme representado na Figura 9d. Sendo assim, se um nó desconhecido estiver fora da região estabelecida pelos nós de refe-

rência, não haverá a possibilidade de acerto na estimativa da posição. Da mesma forma, o erro da posição estimada pelo Min-Max tende a aumentar à medida que a posição real do nó se afasta da região estabelecida pelos nós de referência. É desejável, portanto, que o nó desconhecido esteja dentro das regiões estabelecidas pelos nós de referência, para que sua posição inicial seja estimada com maior precisão.



(a) 5 nós de referência

(b) 3 nós de referência

(c) 2 nós de referência

(d) 1 nó de referência

Figura 9: Regiões em que residem as posições iniciais calculadas pelo Min-Max

Uma alternativa ao uso do Min-Max é o uso de algoritmos de otimização, como o PSO, empregado em (EKBERG, 2009), ou o uso do método da *Lateração* (LANGENDOEN; REIJERS, 2003). Entretanto, mesmo com o uso destes processos, o cálculo da posição inicial continua dependente de $l_{i,r}$, que é predominantemente diferente da distância real entre i e r. Uma vez que ambos os métodos são impactados pela imprecisão de $l_{i,r}$, escolhemos o Min-Max por sua simplicidade, o que representa um menor custo computacional, quando comparado ao método da *Lateração* ou a um algoritmo de otimização.

4.2.3 ETAPA-III: Pré Refinamento

Dependendo da topologia da rede, do tempo de execução do método Sum-Dist, T_S, ou mesmo por falhas de comunicação na ETAPA-I, pode ocorrer que os nós desconhecidos não tomem conhecimento de todos os nós de referência do sistema. Isto pode degradar

a qualidade da posição inicial estimada na ETAPA-II, especialmente nos casos em que um nó desconhecido só é capaz de descobrir menos do que três nós de referência durante a ETAPA-I. Nestes casos, a posição estimada pelo método Min-Max estará contida em uma linha que liga as duas referências, conforme a Figura 9c, ou sobre um único ponto, conforme a Figura 9d. Caso o nó desconhecido não esteja contido nestas regiões, ou em suas imediações, haverá um aumento do erro de posição. Para minimizar este problema, o método MCMM (SÁ; NEDJAH; MOURELLE, 2015c) é proposto, com o intuito de melhorar a qualidade da posição inicial dos nós que tenham conhecido apenas um ou dois nós de referência durante a ETAPA-I.

O método MCMM considera que a amplitude do possível erro de uma posição estimada pelo método Min-Max, numa dada dimensão, tende a diminuir à medida que o comprimento da aresta de S_i na dada dimensão torna-se menor. Assim, a idéia basilar deste método é utilizar a contribuição de todos os nós desconhecidos da rede, para reduzir o tamanho de S_i de nós que conheçam apenas um ou dois nós de referência. Dessa forma, suas fronteiras são redefinidas com base na posição estimada e na distância medida para os nós desconhecidos, que são iterativamente eleitos como novos nós de referência.

Após a execução do método Min-Max, na ETAPA-II, cada nó de referência passa a ter uma posição estimada $u_i = (x_i, y_i)$ e uma área $S_i = \Delta_{x_i} \times \Delta_{y_i}$, onde Δ_{x_i} e Δ_{y_i}, são calculados conforme a Equação 17, e representam o comprimento de S_i nas dimensões x e y, respectivamente:

$$\begin{aligned} \Delta_{x_i} &= \min_{\forall r \in R_i} (\hat{x}_r + l_{i,r}) - \max_{\forall r \in R_i} (\hat{x}_r - l_{i,r}), \\ \Delta_{y_i} &= \min_{\forall r \in R_i} (\hat{y}_r + l_{i,r}) - \max_{\forall r \in R_i} (\hat{y}_r - l_{i,r}). \end{aligned} \tag{17}$$

A cada iteração j do método MCMM, os nós desconhecidos trocam mensagens com o objetivo de compartilhar as informações de x_i, y_i, Δ_{x_i} e Δ_{y_i}. Após a troca de mensagens, o nó desconhecido com o menor Δ_x é eleito como uma nova referência ϵ_x na dimensão x, desde que o mesmo ainda não tenha sido eleito nesta dimensão. Da mesma forma, o nó desconhecido com o menor Δ_y é eleito como uma nova referência ϵ_y na dimensão y, desde que o mesmo também não tenha sido eleito nesta dimensão. Uma vez eleitos ϵ_x e ϵ_y, o algoritmo Sum-Dist é executado a fim de que todos os nós desconhecidos $i \in I$ possam estimar l_{i,ϵ_x} e l_{i,ϵ_y}, que representam as distâncias entre o nó desconhecido i e as referências eleitas ϵ_x e ϵ_y, respectivamente. Finalmente, todos os nós desconhecidos i atualizam x_i, y_i, Δ_{x_i} e Δ_{y_i} numa dada dimensão, se e somente se:

(i) O nó i conhecer menos do que três nós de referência durante a ETAPA-I; e

(ii) Não for possível verificar um bom alinhamento entre o i e dois nós de referência reais r e r', *i.e.* se a Equação 18 for verdadeira,

$$\frac{l_{i,r} + l_{i,r'}}{\sqrt{(\hat{x}_r - \hat{x}_{r'})^2 + (\hat{y}_r - \hat{y}_{r'})^2}} - 1 \geq \beta, \tag{18}$$

onde r e r' são nós de referência reais, sendo $r \neq r'$, $l_{i,r}$ e $l_{i,r'}$ são as distâncias estimadas entre o nó i e os nós r e r', e β é o limiar de alinhamento ajustado empiricamente.

Um nó desconhecido i, que tenha se baseado em apenas dois nós de referência reais, r e r', para estimar sua posição inicial, pode ter uma boa estimativa de posição caso a posição real de i esteja sobre a reta que liga r e r', e caso as distâncias estimadas $l_{i,r}$ e $l_{i,r'}$ se aproximem das distâncias reais entre estes nós. Assim, o critério definido na Equação 18 visa a verificar o alinhamento do nó i com dois nós de referência distintos, bem como a exatidão de $l_{i,r}$ e $l_{i,r'}$. Caso haja um bom alinhamento e uma boa exatidão de $l_{i,r}$ e $l_{i,r'}$, o nó desconhecido não atualiza x_i, y_i, Δ_{x_i} e Δ_{y_i} durante a aplicação do método MCMM.

Caso as condições (i) e (ii) sejam atendidas, as coordenadas x_i e y_i são atualizadas conforme a Equação 19:

$$\begin{aligned} x_i &= \left(\frac{\min\limits_{\forall k \in R_i \cup E_x} \left(x_k + l_{i,k}\right) + \max\limits_{\forall k \in R_i \cup E_x} \left(x_k - l_{i,k}\right)}{2} \right), \\ y_i &= \left(\frac{\min\limits_{\forall k \in R_i \cup E_y} \left(y_k + l_{i,k}\right) + \max\limits_{\forall k \in R_i \cup E_y} \left(y_k - l_{i,k}\right)}{2} \right), \end{aligned} \tag{19}$$

onde E_x e E_y são o conjunto de nós de referência eleitos nas dimensões x e y, respectivamente, e k pode ser tanto um nó de referência real quanto um nó de referência eleito. Vale lembrar que R_i é o conjunto de nós de referência reais conhecidos pelo nó i na ETAPA-I. Ademais, Δ_{x_i} e Δ_{y_i} são atualizados conforme a Equação 20:

$$\begin{aligned} \Delta_{x_i} &= \min_{\forall k \in (R_i \cup E_x)} (x_k + l_{i,k}) - \max_{\forall k \in (R_i \cup E_x)} (x_k - l_{i,k}), \\ \Delta_{y_i} &= \min_{\forall k \in (R_i \cup E_y)} (y_k + l_{i,k}) - \max_{\forall k \in (R_i \cup E_y)} (y_k - l_{i,k}). \end{aligned} \tag{20}$$

A Figura 10 ilustra o trabalho feito durante uma iteração do método MCMM para aumentar a precisão da posição de um nó desconhecido i, baseando-se na contribuição de

dois nós de referência reais, r_1 e r_2, e dois nós de referência eleitos: ϵ_x na dimensão x, e ϵ_y na dimensão y.

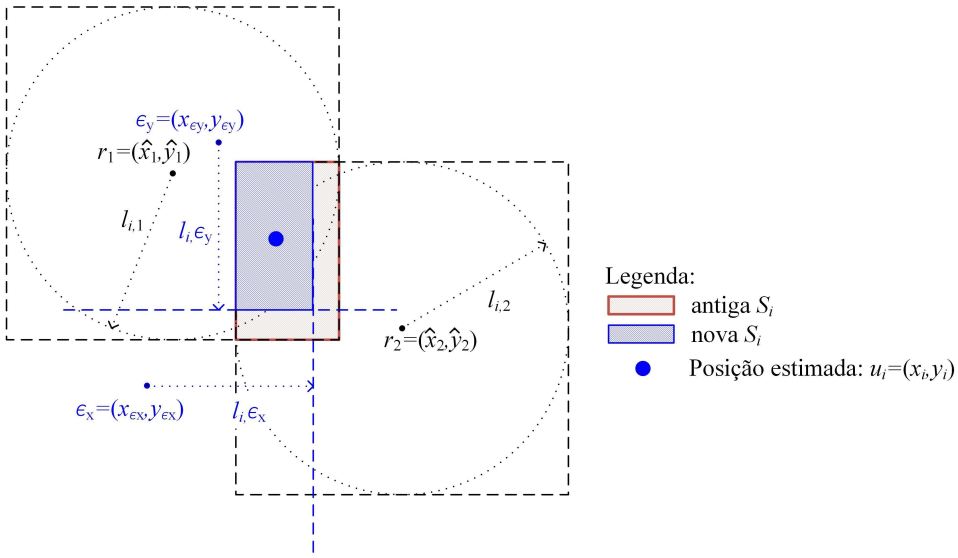


Figura 10: Representação gráfica da aplicação do método MCMM

O procedimento supracitado se repete até que todos os nós desconhecidos tenham sido eleitos como nós de referência em ambas as dimensões. O Algoritmo 5, executado por cada nó desconhecido $i \in I$, descreve os passos do método MCMM, onde n é o número de nós desconhecidos da rede ou enxame.

Algoritmo 5 MCMM

início
  Faça x_i, y_i, Δ_{x_i} e Δ_{y_i}, conforme calculado na ETAPA-II;
  para $j = 1$ *a* n **faça**
    Troca mensagens para compartilhar x_i, y_i, Δ_{x_i} e Δ_{y_i};
    Elege ϵ_x (nó desconhecido, ainda não eleito na dimensão x, com menor Δ_x);
    Elege ϵ_y (nó desconhecido, ainda não eleito na dimensão y, com menor Δ_y);
    Executa o Sum-Dist para estimar l_{i,ϵ_x} e l_{i,ϵ_y};
    se *Condições (i) e (ii) forem atendidas* **então**
      Atualiza x_i e y_i conforme (19);
      Atualiza Δ_{x_i} e Δ_{y_i} conforme (20);
    fim
  fim
  retorna x_i e y_i;
fim

4.2.4 ETAPA-IV: Refinamento das Posições

Após o pré-refinamento das posições de cada nó desconhecido, na ETAPA-III, é realizada a ETAPA-IV, que visa ao refinamento das posições. Nesta etapa, a cada ciclo iterativo, cada nó desconhecido recalcula sua posição com base nas posições de seus vizinhos, que também são atualizadas ciclo a ciclo, e nas distâncias medidas para os mesmos. Uma vez que, idealmente, as medidas de distância entre nós permanecem constantes, as posições estimadas pelos nós tendem a se ajustar gradativamente, de forma a convergir para as posições reais. O refinamento é obtido por meio de um processo de otimização que visa a minimizar a função objetivo descrita na Seção 4.2.4.1, a qual muda dinamicamente à medida que a vizinhança do nó atualiza suas posições. Duas alternativas de algoritmo de otimização são utilizadas nesta etapa: o PSO e o BSA, descritos nas Seções 3.2 e 3.3, respectivamente. A forma como os algoritmos de otimização são aplicados ao problema dinâmico em questão é apresentada na Seção 4.2.4.2.

4.2.4.1 A Função Objetivo

Na solução proposta nesta dissertação, o PSO e o BSA são utilizados para minimizar uma função objetivo composta de duas parcelas. Para definir a primeira parcela, é preciso considerar que, para um dado nó desconhecido i, o erro em distância para cada nó vizinho $v \in V_i$ é definido conforme a Equação 21 (EKBERG, 2009):

$$g_{i,v} = d_{i,v} - ||p_v - p_i||, \tag{21}$$

onde $d_{i,v}$ é a distância medida entre o nó desconhecido i e o nó vizinho v, por um dos métodos descritos na Seção 1.2, p_v é a posição inferida por v e p_i é a posição estimada para i. Neste caso, V_i é o conjunto de nós vizinhos v que estão a um salto de distância de i, seja v um nó de referência ou não. A Figura 11 ilustra o cálculo de $g_{i,v}$ feito para um indivíduo (ou partícula) do algoritmo de otimização, considerando a existência de três vizinhos a um salto de distância. Note que, no exemplo dado, onde não há erro de medidas de distância, quando todos $g_{i,v}$ forem iguais a zero, o indivíduo terá as coordenadas correspondentes à posição real do nó, *i.e.* a interseção das circunferências cujos raios são as distâncias medidas diretamente para os vizinhos $v \in V_i$. Assim, a primeira parcela da função objetivo é obtida pelo somatório do quadrado de $g_{i,v}$ dividido por um fator de confiança, que é

atribuído a cada nó vizinho. Dessa, forma, tem-se a Equação 22:

$$f_1 = \sum_{v \in V_i} \frac{{g_{i,v}}^2}{\zeta_v}, \tag{22}$$

onde ζ_v, a ser definido na Seção 4.2.5, é o fator de confiança atribuído ao nó vizinho v.

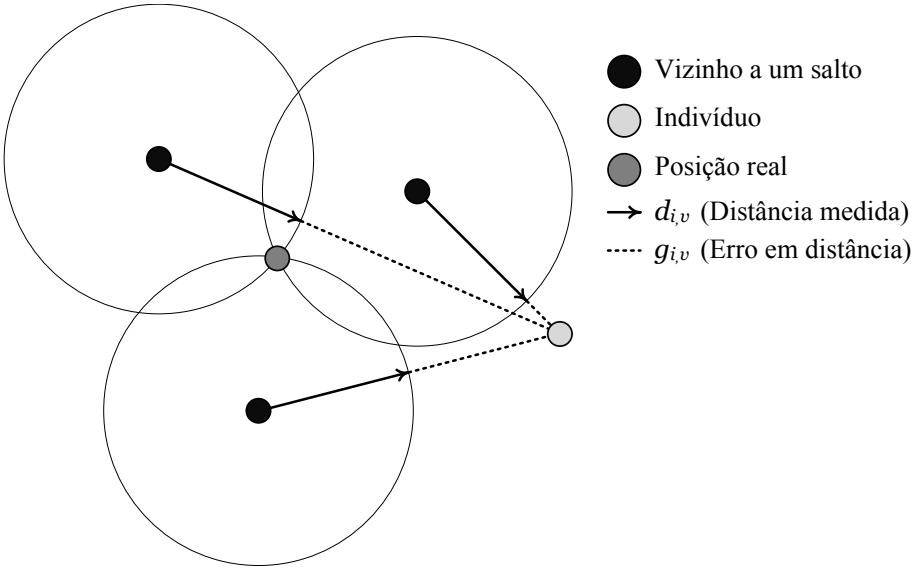


Figura 11: Cálculo do erro em distância $g_{i,v}$ para nós vizinhos a um salto

A segunda parcela da função objetivo é calculada com base em nós que estão a dois saltos de distância do nó desconhecido i. Segundo (EKBERG, 2009), é possível afirmar que a distância entre o nó desconhecido i e um nó w, a dois saltos de distância de i, será sempre maior que a distância limite de medição L e menor do que $2L$. Dessa forma, a contribuição do um nó w para a segunda parcela da função de aptidão é calculada conforme a Equação 23 (EKBERG, 2009):

$$h_{i,w} = \max(0, L - ||p_w - p_i||, ||p_w - p_i|| - 2L)^2, \tag{23}$$

onde p_w é a posição do nó w.

Observe que, se uma possível solução for tal que $L \leq ||p_w - p_i|| \leq 2L$, nenhum valor será somado à sua aptidão. Se $||p_w - p_i|| < L$, será somado à função de aptidão um valor igual ao quadrado da distância entre p_i e a circunferência de raio L, com centro em p_w. Se $2L < ||p_w - p_i||$, será somado à função de aptidão um valor igual ao quadrado da distância entre p_i e a circunferência de raio $2L$, com centro em p_w. A Figura 12 representa graficamente o cálculo de $h_{i,w}$.

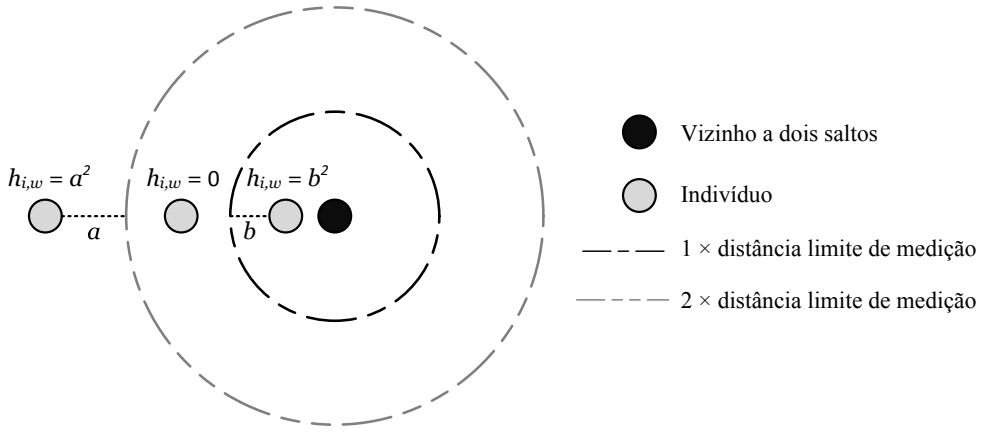


Figura 12: Cálculo de $h_{i,w}$ para um nó vizinho a dois saltos

Ante o exposto, a segunda parcela da função objetivo é calculada conforme a Equação 24 (EKBERG, 2009):

$$f_2 = \sum_{w \in W_i} \frac{h_{i,w}}{\zeta_w}, \tag{24}$$

onde W_i é o conjunto de nós que estão a dois saltos de distância de i e ζ_w, calculado conforme a ser descrito na Seção 4.2.5, é o fator de confiança do nó w.

Por fim, a função objetivo, que deve ser minimizada, é então definida pela Equação 25:

$$\min_{p_i \in A} f = f_1 + f_2, \tag{25}$$

onde p_i é a posição do nó i e A é todo o espaço de busca. Note que, se considerarmos um sistema ideal, onde não há erros de medida de distância, o valor mínimo de f é 0.

4.2.4.2 Otimização do Problema Dinâmico

A função objetivo, modelada na Seção 4.2.4.1 para encontrar a posição do nó desconhecido i, leva em consideração as posições dos nós v e w, vizinhos a um e dois saltos de distância de i, respectivamente. Tais nós podem ser tanto nós desconhecidos quanto nós de referência. Considerando a possibilidade de um ou mais nós vizinhos, v ou w, serem nós desconhecidos, estes também passarão pelo processo de refinamento, simultaneamente com o nó i. Sendo assim, as posições estimadas pelos nós desconhecidos v ou w mudam dinamicamente à medida que a etapa de refinamento é executada, fazendo necessária a atualização da função objetivo no nó i. Isto resulta em um problema dinâmico, onde a função objetivo muda no decorrer do processo de otimização.

Uma solução para o problema dinâmico seria simplesmente reiniciar o algoritmo de otimização a cada mudança na função objetivo. Entretanto, isto levaria a uma perda da evolução obtida pelo enxame até o momento. Consideramos, assim como (EKBERG, 2009), ser provável que, a cada mudança, a função objetivo mantenha alguma semelhança com a sua forma anterior, fazendo com que as posições assumidas pelo enxame ainda tenha algum valor. Sendo assim, para lidar com o problema dinâmico, os algoritmos de otimização, PSO e BSA, foram aplicados seguindo o fluxograma apresentado na Figura 13. O parâmetro Δ define o número de ciclos iterativos da ETAPA-IV, enquanto que δ define o número de iterações do algoritmo de otimização (PSO ou BSA), executadas durante cada ciclo da ETAPA-IV. Note que a cada ciclo iterativo da ETAPA-IV, a função objetivo, definida na Seção 4.2.4.1, é atualizada com as novas posições informadas pela vizinhança do nó i, atendendo à característica dinâmica do problema. O enxame é inicializado apenas na primeira iteração da ETAPA-IV. Nas iterações subsequentes, o enxame é preservado para que as posições adquiridas até o momento sejam aproveitadas. No entanto, após a atualização da função objetivo, o enxame é reavaliado de acordo com a função objetivo atualizada. No caso do PSO, durante a avaliação do enxame, a melhor posição global m_g e a melhor posição da partícula m_ℓ são atualizadas. A melhor posição da partícula recebe a sua posição atual, *i.e.* $m_\ell \leftarrow \chi_\ell$, e a melhor posição global m_g recebe a posição da melhor partícula, segundo as novas aptidões. No caso do BSA, apenas as aptidões da população atual P são atualizadas.

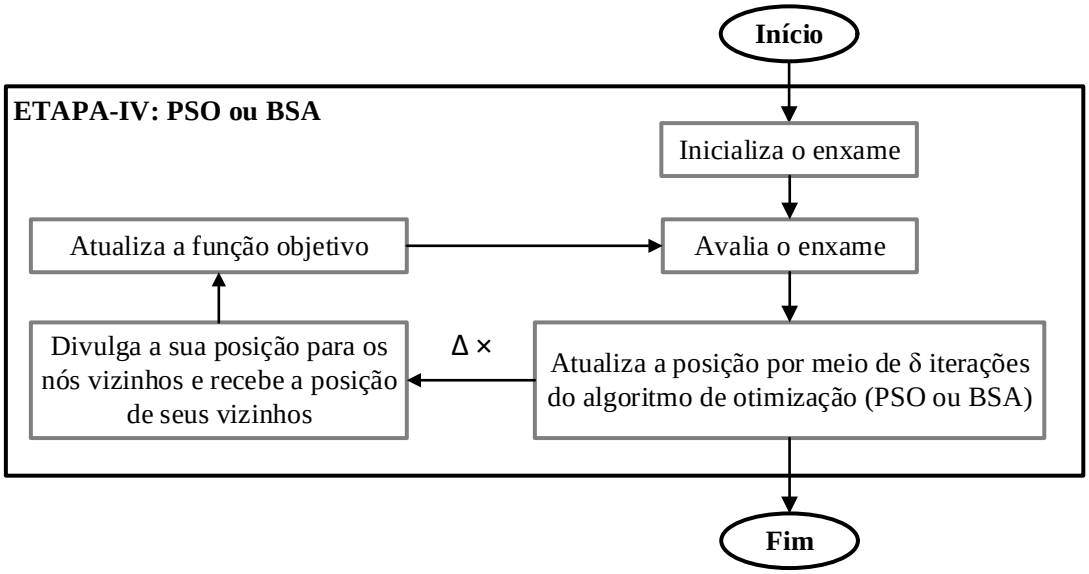


Figura 13: Fluxograma da ETAPA-IV

4.2.5 Fator de Confiança

O fator de confiança visa a dar maior importância às contribuições dadas por nós que tendam a ter posições mais acuradas, sejam eles vizinhos a um salto (vide Equação 22) ou a dois saltos (vide Equação 24). Dois métodos distintos foram empregados para calcular o fator de confiança. Em ambos os métodos, o fator de confiança atribuído aos nós de referência é igual a 1.

O primeiro método (EKBERG, 2009), referenciado nesta dissertação por Fator de Confiança por Saltos às Referências, ou HTA-CF, se baseia na idéia de que os nós que estão mais próximos dos nós de referência tendem a ter a posição mais precisa. Neste método, os fatores de confiança $\zeta_i^{(\mathrm{HTA})}$, de um nó desconhecido i, e $\zeta_r^{(\mathrm{HTA})}$, de um nó de referência r, são definidos pela Equação 26:

$$\begin{aligned} \zeta_i^{(\mathrm{HTA})} &= \sum_{r \in R'_i} \lambda_{i,r}^2, \\ \zeta_r^{(\mathrm{HTA})} &= 1, \end{aligned} \tag{26}$$

onde R'_i é o conjunto formado pelos três nós de referência mais próximos a i, em números de saltos, e $\lambda_{i,r}$ é o número de saltos entre o nó i e o nó de referência r.

O segundo método (SÁ; NEDJAH; MOURELLE, 2014b), ora proposto, designado como Fator de Confiança pela Área Min-Max, ou MMA-CF, leva em consideração a área $S_i = \Delta_{x_i} \times \Delta_{y_i}$ estabelecida pelo método Min-Max, a qual, na ausência de erros de medida de distância, corresponde a uma região onde reside o nó desconhecido i (SAVVIDES; PARK; SRIVASTAVA, 2002). Sendo assim, pode-se inferir que, quanto menor for esta região, maior será a possibilidade da posição u_i, estabelecida pelo método Min-Max, estar perto da posição real.

Ocorre que, dependendo da topologia da rede, um nó i afastado dos nós de referência pode ter uma pequena área S_i e, portanto, ter boa confiabilidade. Isto acontece, por exemplo, em casos em que há um bom alinhamento dos nós que estão entre o nó i e os nós de referência. Nestes casos, as distâncias $l_{i,r}$ se aproximam das distâncias reais entre i e $r \in R$, reduzindo o tamanho de S_i. No método MMA-CF, os fatores de confiança $\zeta_i^{(\mathrm{MMA})}$, de um nó desconhecido i, e $\zeta_r^{(\mathrm{MMA})}$, de um nó de referência r, são definidos pela Equação 27:

$$\begin{aligned} \zeta_i^{(\mathrm{MMA})} &= 1 + |S_i|, \\ \zeta_r^{(\mathrm{MMA})} &= 1. \end{aligned} \tag{27}$$

A Figura 14 permite comparar os fatores de confiança calculados pelos métodos HTA-CF e MMA-CF para uma rede com 10 nós de referência e 200 nós desconhecidos, aleatoriamente distribuídos. Nesta figura, os nós foram classificados em ordem crescente em função do fator de confiança calculado pelo método HTA-CF. É possível observar que alguns nós que possuem alto fator de confiança pelo método HTA-CF, *i.e.* estão afastados dos nós de referência da rede, apresentam baixo fator de confiança pelo método MMA-CF, *i.e.* possuem uma pequena área S_i. A título de exemplo, alguns destes nós foram destacados na Figura 14.

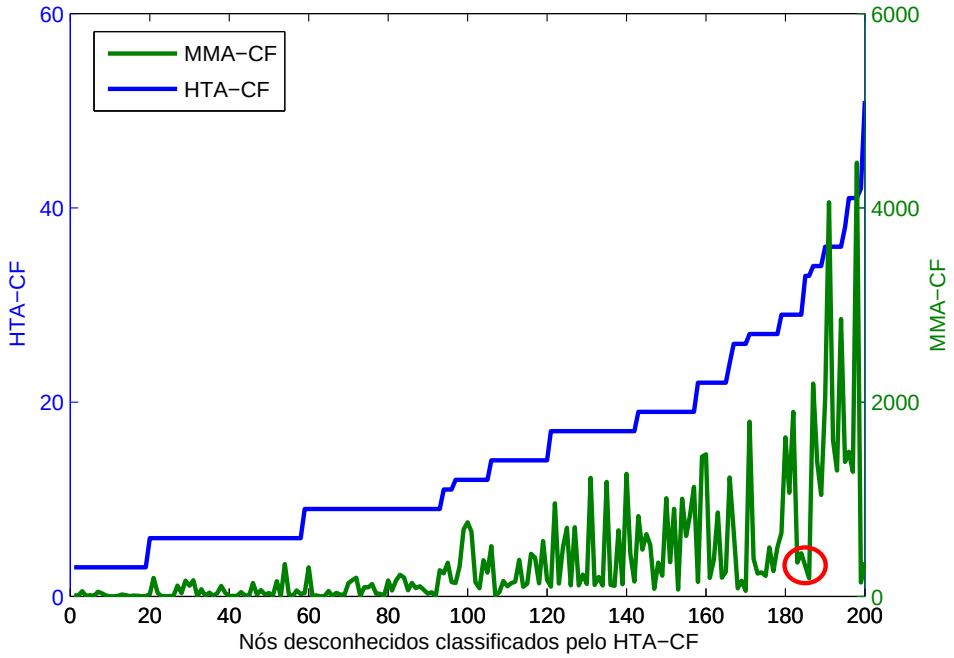


Figura 14: Comparação entre o HTA-CF e o MMA-CF

Isto indica que, mesmo estando afastados dos nós de referência, estes nós tendem a ter uma boa acurácia na posição inicial, calculada pelo método Min-Max, e podem ser considerados, ao menos nos primeiros ciclos da ETAPA-IV, nós de boa confiabilidade, mesmo estando longe dos nós de referência. Desta forma, o uso do o método MMA-CF sugere a possibilidade de melhoria no desempenho do processo de otimização executado na ETAPA-IV.

Cabe ressaltar que, conforme descrito na Seção 4.2.3, podem ocorrer situações em que a posição inicial do nó desconhecido i e sua área S_i, estabelecidas na ETAPA-II, sejam comprometidas em virtude eventuais dificuldades deste nó obter dados dos nós de

referência. Nestes casos, o cálculo da Equação 27 é feito com base na área S_i computada com os valores de Δ_{x_i} e Δ_{y_i} atualizados pelo MCMM. Os resultados obtidos pelos dois métodos de cálculo do fator de confiança são apresentados e comparados no Capítulo 5.

4.2.6 Algoritmos Completos para o Método Proposto

Dois algoritmos foram desenvolvidos com base no método proposto: o Algoritmo 6 (SÁ; NEDJAH; MOURELLE, 2014a) e o Algoritmo 7 (SÁ; NEDJAH; MOURELLE, 2014b). Ambos os algoritmos reúnem as quatro etapas apresentadas nas Seções 4.2.1, 4.2.2, 4.2.3 e 4.2.4. No Algoritmo 6, o PSO é utilizado para fazer o refinamento das posições, enquanto que no Algoritmo 7, o BSA é utilizado para realizar tal refinamento. Note que o mesmo algoritmo é executado simultaneamente por cada nó da rede, fazendo com que a solução emirja de um processamento distribuído.

Nas ETAPAS-I e II, o algoritmo executa os cálculos referentes aos métodos Sum-Dist e Min-Max para obter a posição inicial u_i, que irá subsidiar o início da ETAPA-III, de pré-refinamento. Na ETAPA-III, é executado o método MCMM, conforme o procedimento descrito no Algoritmo 5. Ao término do método MCMM, são trocadas as informações de p_i, a fim de que cada nó reúna os dados de p_v e p_w, que serão utilizados na função objetivo da ETAPA-IV. Ainda na ETAPA-III, cada nó desconhecido i calcula o seu fator de confiança ζ_i, de acordo com as Equações 26 ou 27. Da mesma forma, as informações de ζ_i são trocadas entre vizinhos a fim de que cada nó reúna os dados de ζ_v e ζ_w, que também serão utilizados na função objetivo da ETAPA-IV.

Na ETAPA-VI, ao iniciar o algoritmo de otimização (PSO ou BSA), a posição p_i, calculada na ETAPA-III, é inserida no enxame (ou população). Em ambos os algoritmos, cada nó minimiza uma função objetivo que agrega as informações colhidas de sua vizinhança, conforme definido na Seção 4.2.4.1, por meio de δ iterações do algoritmo de otimização (PSO ou BSA). Em seguida, os nós da rede trocam mensagens contendo suas novas posições estimadas. As novas posições p_v e p_w, informadas pelos nós vizinhos, são utilizadas para atualizar a função objetivo. Uma vez atualizada a função objetivo, a mesma é novamente minimizada pelo algoritmo de otimização. Este processo se repete até que a condição de parada seja atendida, *i.e.* Δ ciclos da ETAPA-IV. No caso do Algoritmo 6, o número de particulas do PSO é definido por Y.

Algoritmo 6 Algoritmo completo no nó i, utilizando o PSO na ETAPA-IV

início
  ETAPA-I: Sum-Dist
    Troca mensagens para obter $(\hat{x}_r, \hat{y}_r, l_{i,r})$ de $r \in R_i$, conforme Algoritmo 4;
  fim
  ETAPA-II: Min-Max
    Calcula u_i usando o método Min-Max;
  fim
  ETAPA-III: MCMM
    Executa o método MCMM, conforme o Algoritmo 5;
    Troca mensagens para compartilhar p_i com todo $v \in V_i$ e $w \in W_i$;
    Calcula o fator de confiança ζ_i;
    Troca mensagens para compartilhar ζ_i com todo $v \in V_i$ e $w \in W_i$;
  fim
  ETAPA-IV: refinamento com o PSO
    Inicia o enxame com posições χ_ℓ e velocidades ν_ℓ aleatórias;
    Insere p_i no enxame;
    Avalia o enxame considerando (p_v, ζ_v) $\forall v \in V_i$ e (p_w, ζ_w) $\forall w \in W_i$;
    Armazena m_i e m_g;
    para $1 \leq j \leq \Delta$ **faça**
      se *recebidas atualizações de* p_v *e* p_w **então**
        Atualiza função objetivo com os p_v e p_w recebidos na ETAPA-IV;
        para *toda partícula* ℓ, $1 \leq \ell \leq Y$ **faça**
          Reavalia χ_ℓ;
          $m_\ell \leftarrow \chi_\ell$;
          $m_g \leftarrow \chi_\ell$, se $f(\chi_\ell) < f(m_g)$;
        fim
      fim
      para $1 \leq k \leq \delta$ **faça**
        para *toda partícula* ℓ, $1 \leq \ell \leq Y$ **faça**
          Atualiza velocidade ν_ℓ, conforme as Equações 8 e 9;
          Atualiza posição χ_ℓ, conforme a Equação 7;
          aptidão $\leftarrow f(\chi_\ell)$;
          $m_\ell \leftarrow \chi_\ell$, se *aptidão* $< f(m_\ell)$;
          $m_g \leftarrow \chi_\ell$, se *aptidão* $< f(m_g)$;
        fim
      fim
      $p_i \leftarrow m_g$;
      Troca mensagens para compartilhar as novas p_i;
    fim
    retorna p_i;
  fim
fim

Algoritmo 7 Algoritmo completo no nó i, utilizando o BSA na ETAPA-IV

```
início
    ETAPA-I: Sum-Dist
        Troca mensagens para obter (x̂_r, ŷ_r, l_{i,r}) de r ∈ R_i, conforme Algoritmo 4;
    fim
    ETAPA-II: Min-Max
        Calcula u_i usando o método Min-Max;
    fim
    ETAPA-III: MCMM
        Executa o método MCMM, conforme o Algoritmo 5;
        Troca mensagens para compartilhar p_i com todo v ∈ V_i e w ∈ W_i;
        Calcula o fator de confiança ζ_i;
        Troca mensagens para compartilhar ζ_i com todo v ∈ V_i e w ∈ W_i;
    fim
    ETAPA-IV: refinamento com o BSA
        Inicializa P e P_hist;
        Insere p_i em P;
        Calcula a aptidão de P usando (p_v, ζ_v) ∀v ∈ V_i e (p_w, ζ_w) ∀w ∈ W_i;
        para 1 ≤ j ≤ Δ faça
            se recebidas atualizações de p_v e p_w então
                Atualiza função objetivo com os p_v e p_w recebidos na ETAPA-IV;
                Reavalia P;
            fim
            para 1 ≤ k ≤ δ faça
                Seleção-I;
                Gera nova população
                    Mutação;
                    Cruzamento;
                fim
                Seleção-II;
            fim
            Troca mensagens para compartilhar as novas p_i;
        fim
    fim
    retorna p_i;
fim
```

4.3 Considerações Finais

Este capítulo apresentou uma solução para o problema de localização em múltiplos saltos, a qual é estruturada em quatro etapas. Na escolha dos métodos adotados em cada etapa, buscou-se um algoritmo que pudesse ser implementado em dispositivos com recursos de *hardware* limitados, como é o caso dos robôs do tipo *Kilobot*, empregados nos experimentos desta dissertação. A solução, baseada em medidas de distância, é adequada para os

referidos robôs, tendo em vista o fato destes serem dotados de sensores de infravermelho capazes de medir a distância entre robôs vizinhos.

Na solução apresentada, a abordagem de três etapas proposta por (LANGENDOEN; REIJERS, 2003), é estendida para uma nova abordagem de quatro etapas. Isto decorre da inserção de um novo método responsável por executar a etapa de pré-refinamento, o método MCMM. A solução introduz ainda um novo método para determinar o fator de confiança de um nó, denominado MMA-CF, que passa a ser uma alternativa ao método HTA-CF, proposto por (EKBERG, 2009).

Com base na solução proposta, dois algoritmos foram desenvolvidos, diferindo apenas no algoritmo de otimização empregado para fazer o refinamento das posições: um algoritmo utiliza o PSO, enquanto o outro faz uso do BSA. O desempenho dos métodos Min-Max e MCMM, bem como os resultados alcançados ao final dos dois algoritmos apresentados, utilizando ambos os fatores de confiança HTA-CF e MMA-CF, são avaliados no Capítulo 5.

Capítulo 5

RESULTADOS

NESTE capítulo, são apresentados e analisados os resultados alcançados com o método descrito no Capítulo 4. A avaliação do método proposto é feita por meio de simulações conduzidas no MATLAB® (MATHWORKS, 2005), cujos resultados obtidos são apresentados na Seção 5.1, e experimentos realizados em robôs do tipo *Kilobot* (RUBENSTEIN; AHLER; NAGPAL, 2012), cujos resultados constam na Seção 5.2.

5.1 Simulação

Conforme descrito na Seção 4.2.4, foram consideradas duas alternativas de algoritmos de otimização para a ETAPA-IV: o PSO e o BSA. Da mesma forma, foram considerados dois métodos alternativos para calcular o fator de confiança dos nós da rede: o HTA-CF e o MMA-CF. Sendo assim, nesta seção, são avaliados quatro algoritmos resultantes da combinação das referidas alternativas, os quais, tal como em (SÁ; NEDJAH; MOURELLE, 2015a), diferem apenas na ETAPA-IV:

1. BSA-HTA: em que a ETAPA-IV se baseia no BSA e o fator de confiança dos nós é calculado pelo método HTA-CF;

2. BSA-MMA: em que a ETAPA-IV se baseia no BSA e o fator de confiança dos nós é calculado pelo método MMA-CF;

3. PSO-HTA: em que a ETAPA-IV se baseia no PSO e o fator de confiança dos nós é calculado pelo método HTA-CF;

4. PSO-MMA: em que a ETAPA-IV se baseia no PSO e o fator de confiança dos nós é calculado pelo método MMA-CF;

Como base para as versões BSA-HTA e BSA-MMA, utilizou-se a implementação do BSA disponível em (CIVICIOGLU, 2013b). O parâmetro K utilizado no BSA foi ajustado empiricamente por meio de uma série de simulações. Os melhores resultados foram obtidos para $K = 1$. Do mesmo modo, os parâmetros do PSO foram ajustados empiricamente, de forma que os coeficientes que obtiveram melhores resultados são apresentados na Tabela 1. Em todas as simulações, tanto o PSO quanto o BSA foram empregados com uma população de 100 indivíduos. Os valores de Δ e δ, da ETAPA-IV, foram ajustados para 40 e 10, respectivamente. O limiar de alinhamento do MCMM foi ajustado para $\beta = 0,006$.

Tabela 1: Parâmetros do PSO

Parâmetros	Valor
ω	0,73
φ_1	2,05
φ_2	2,05
δ	0,01

Primeiramente, na Seção 5.1.1, é feita uma comparação dos resultados obtidos pelos quatro algoritmos propostos, com os resultados alcançados pelos algoritmos *Robust Positioning* (RABAEY; LANGENDOEN, 2002) e *N-hop Multilateration* (SAVVIDES; PARK; SRIVASTAVA, 2002) em (LANGENDOEN; REIJERS, 2003). Os algoritmos *Robust Positioning* e *N-hop Multilateration* foram adotados como referência, para comparação, em virtude de serem reconhecidamente robustos (LANGENDOEN; REIJERS, 2003) e estruturados em etapas similares aos algoritmos ora propostos. Posteriormente, na Seção 5.1.2, é feita uma análise do desempenho do método proposto em face de diferentes quantidades de nós de referência e de nós desconhecidos. Na Seção 5.1.3, é avaliado o desempenho dos quatro algoritmos em diferentes topologias. Cabe ressaltar que nas simulações apresentadas nas Seções 5.1.1, 5.1.2 e 5.1.3, não houve influência do MCMM no processo de localização. Nestas simulações, todos os nós desconhecidos utilizaram mais do que dois nós de referência para estimar sua posição inicial, não atendendo à condição *(i)* do Algoritmo 5, definida na Seção 4.2.3. A melhoria obtida pelo método MCMM é avaliada na Seção 5.1.4, onde são consideradas situações em que os nós desconhecidos utilizam menos do que três nós de referência no cálculo da posição inicial da ETAPA-II.

5.1.1 Comparação com Outros Algoritmos

Em (LANGENDOEN; REIJERS, 2003), o autor avalia e compara, por meio de simulações, os resultados obtidos pelos algoritmos *Robust Positioning* (RABAEY; LANGENDOEN, 2002) e *N-hop Multilateration* (SAVVIDES; PARK; SRIVASTAVA, 2002) na localização de nós em redes de sensores sem fio. Sendo assim, para estabelecer uma comparação de desempenho dos quatro algoritmos propostos em relação aos algoritmos *Robust Positioning* e *N-hop Multilateration*, foram realizadas simulações em 100 cenários seguindo as mesmas especificações utilizadas por (LANGENDOEN; REIJERS, 2003).

Cada cenário é definido em uma área quadrada $A = 100 \times 100$ unidades de medida (u.m.). Inicialmente, 225 nós são distribuídos aleatoriamente, utilizando uma distribuição uniforme, por toda a área em questão. Em seguida, 5% destes nós, *i.e.* 11 nós, são designados como nós de referência. Para escolher os nós de referência, uma malha quadrada com 3×3 vértices é sobreposta em cima da área em questão para que os nós mais próximos dos vértices, *i.e.* 9 nós, sejam designados como nós de referência. Os dois nós de referência remanescentes são escolhidos aleatoriamente, totalizando 11 nós de referência, restando, ao final do processo 214 nós desconhecidos. A Figura 15 apresenta um exemplo de rede gerada segundo os critérios apresentados.

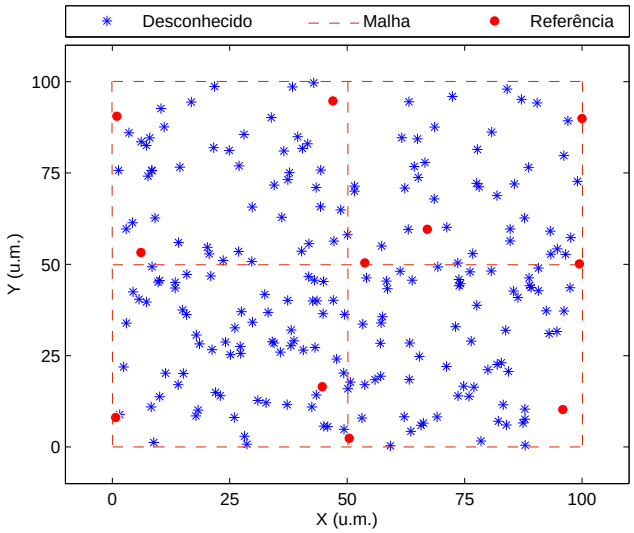


Figura 15: Exemplo de um cenário gerado segundo os critérios usados por (LANGENDOEN; REIJERS, 2003)

Segundo (LANGENDOEN; REIJERS, 2003), a maioria dos algoritmos de localização são bastante sensíveis à presença ou ausência de nós de referência nas extremidades da

rede, sendo este o motivo dos critérios usados na escolha dos nós de referência. Em geral, um nó desconhecido situado nas extremidades de uma rede encontra maior dificuldade para se localizar devido à menor conectividade existente nestas regiões. Entretanto, ao circundar os nós desconhecidos com nós de referência, alguns algoritmos de localização, como o de *Lateração*, por exemplo, apresentam melhor desempenho. Embora o posicionamento específico dos nós de referência possa não ser viável na prática, a maioria dos nós desconhecidos de uma rede de grande escala (redes com mais de 1000 nós) são normalmente cercados por nós de referência (LANGENDOEN; REIJERS, 2003).

Assim como em (LANGENDOEN; REIJERS, 2003), utilizou-se como base o modelo de erro de medida de distância apresentado em (WHITEHOUSE; CULLER, 2002), em que a medida de distância entre nós diretamente conectados é calculada conforme a Equação 28:

$$\hat{d}_{i,v} \sim N\left(d_{i,v}, \sigma\right), \tag{28}$$

onde $\hat{d}_{i,v}$ é a distância medida entre um nó desconhecido i e seu vizinho v, a um salto de distância, e N é uma distribuição normal, onde a média é a distância real $d_{i,v}$ e a variância σ é parametrizada como um percentual de $d_{i,v}$. Da mesma forma que em (LANGENDOEN; REIJERS, 2003) a distância limite de medição foi ajustada para $L = 14$u.m.

Para cada simulação realizada em um cenário c, foi calculada a média do erro de posicionamento de todos os nós desconhecidos, MEP_c, conforme descrito na Equação 29:

$$MEP_c = \frac{\sum_{i=1}^{I} ||p_{real}(i) - p_{calc}(i)||}{I}, \tag{29}$$

onde p_{real} é a posição real do nó e p_{calc} é a posição estimada pelo algoritmo. Vale lembrar que i representa o nó desconhecido e I é o número total de nós desconhecidos.

Para sintetizar as MEP_c de todos os cenários avaliados, foi calculada a $MMEP$, que representa a média das MEP_c, conforme a Equação 30:

$$MMEP = \frac{\sum_{c=1}^{C'} MEP_c}{C'}, \tag{30}$$

tendo, neste caso, o número de cenários avaliados $C' = 100$.

Os resultados obtidos pelos quatro algoritmos propostos, considerando erros na medida de distância, são apresentados nas Figuras 16 e 17. Essas figuras apresentam a $MMEP$ normalizada em relação à distância limite de medição L, *i.e.* um erro de posição

de 0,5 significa que a posição estimada está $0,5L$ afastada da posição real, considerando variâncias de 0 a 0,1. As $MMEPs$ normalizadas das Figuras 16 e 17 são apresentadas com os seus Intervalos de Confiança (IC) de 95%. Na Figura 16 os resultados dos quatro algoritmos propostos são comparados com aqueles obtidos pelo algoritmo *Robust Positioning*, descrito na Seção 2.1.2, o qual utiliza os métodos DV-HOP e *Lateração* nas ETAPAS-I e II, respectivamente, bem como a *Lateração* para realizar o refinamento. Nesta figura, RPR e RP representam os resultados do algoritmo *Robust Positioning* com e sem refinamento, respectivamente. Da mesma forma, SDMM representa os resultados obtidos ao final da ETAPA-II dos quatro algoritmos propostos nesta dissertação, sem refinamento, *i.e.* apenas com o uso dos algoritmos Sum-Dist e Min-Max. A Figura 17 compara os mesmos resultados obtidos pelos quatro algoritmos propostos com aqueles obtidos pelo algoritmo *N-hop Multilateration* (SAVVIDES; PARK; SRIVASTAVA, 2002), que emprega os métodos Sum-Dist e Min-Max nas ETAPAS-I e II, respectivamente, bem como a *Lateração* para realizar o refinamento. Nesta figura, NHMR e NHM representam os resultados do algoritmo *N-hop Multilateration* com e sem refinamento, respectivamente. Em ambas as Figuras 16 e 17, os resultados de RP, RPR, NHM e NHMR foram obtidos de (LANGENDOEN; REIJERS, 2003).

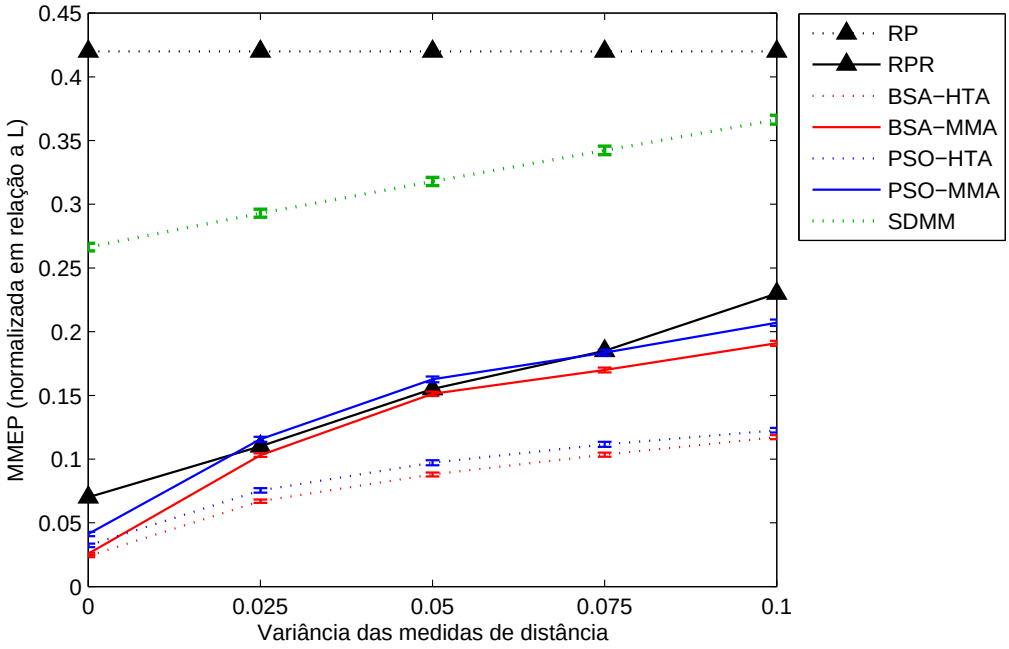


Figura 16: Comparação com o algoritmo *Robust Positioning*

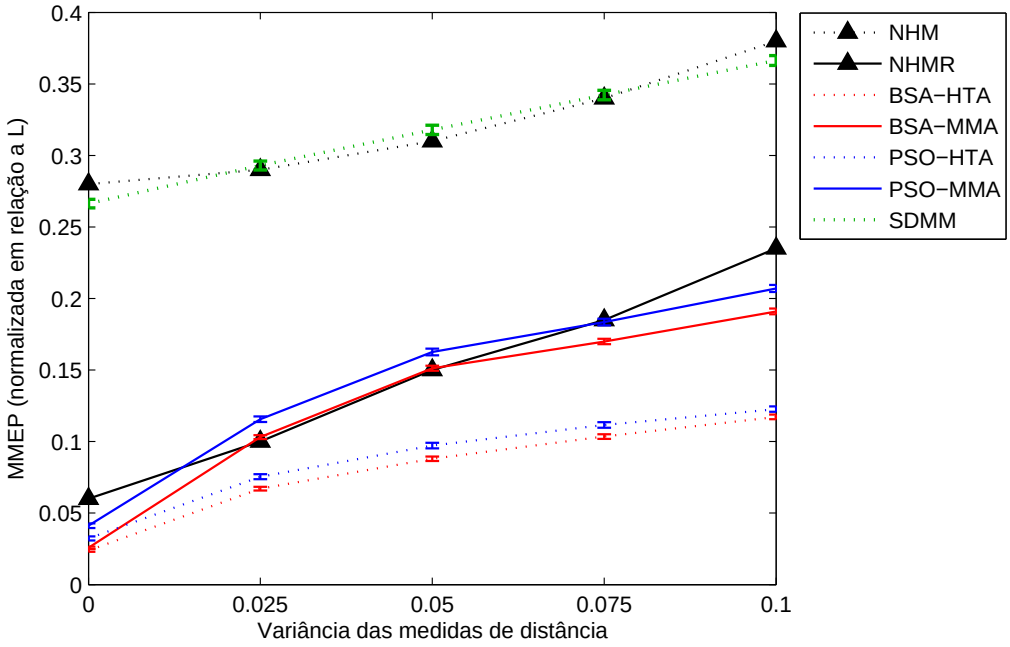


Figura 17: Comparação com o algoritmo *N-hop Multilateration*

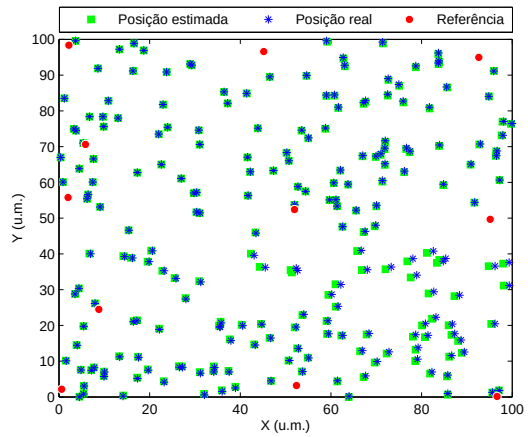
Primeiramente, cabe ressaltar que, de acordo com a Figura 17, a curva SDMM, que representa o resultado obtido ao final da ETAPA-II dos algoritmos propostos, *i.e.* considerando apenas os métodos Sum-Dist e Min-Max, se aproxima do resultado obtido pelos mesmos métodos em (LANGENDOEN; REIJERS, 2003), representado pela curva NHM. Isto sugere que os cenários gerados, apesar de aleatórios, são equivalentes aos utilizados por (LANGENDOEN; REIJERS, 2003).

Com referência aos algoritmos de otimização empregados, o BSA obteve um desempenho ligeiramente melhor do que o PSO, em todas as variâncias de medida de distância, tanto com o fator de confiança HTA-CF, quanto com o MMA-CF. Essa superioridade, entretanto, não excedeu a uma diferença de 0,017 entre as $MMEPs$ normalizadas. No que tange ao fator de confiança, para esta configuração de rede, o HTA-CF obteve um melhor desempenho do que o MMA-CF, considerando um mesmo algoritmo de otimização.

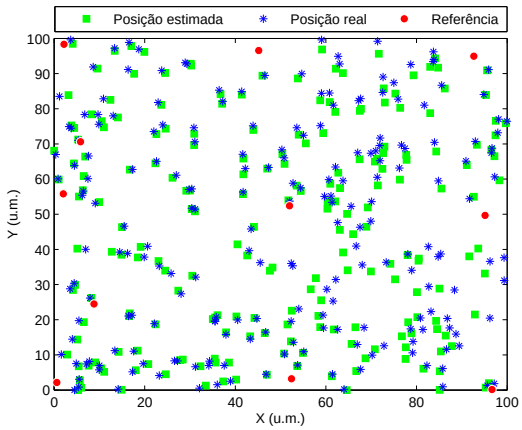
É possível constatar pelas Figuras 16 e 17 a eficácia dos algoritmos propostos, mesmo na presença de erros de medição de distância. De uma forma geral, os quatro algoritmos propostos alcançaram resultados melhores do que os algoritmos RPR e NHMR. À exceção, destacamos o PSO-MMA que, para variâncias de medida de distância de 0,025 e 0,05, obteve desempenho inferior aos algoritmos RPR e NHMR, bem como o BSA-MMA que, para as mesmas variâncias, obteve desempenho inferior ao algoritmo NHMR.

Nesta configuração de rede, o BSA-HTA foi o algoritmo que alcançou melhores

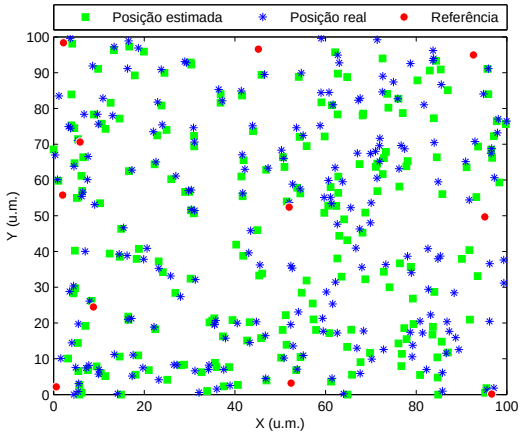
resultados, considerando todas as variâncias de medida de distância. Na melhor condição de erro de medida de distância, *i.e.* com 0 de variância, o BSA-HTA alcançou uma MMEP normalizada de 0,0239, 40% inferior às MMEPs normalizadas obtidas pelos algoritmos RPR e NHMR, que foram em torno de 0,06. Na pior condição de erro de medida de distância, *i.e.* com a variância de 0,1, o BSA-HTA alcançou uma MMEP normalizada de 0,1172, 51% inferior às MMEPs normalizadas obtidas pelos algoritmos RPR e NHMR, que foram de aproximadamente 0,23. A título de exemplo, a Figura 18 apresenta os resultados obtidos pelo BSA-HTA em um mesmo cenário, considerando diferentes variâncias de medida de distância.



(a) 0 de variância



(b) 0,05 de variância



(c) 0,10 de variância

Figura 18: Resultado obtido pelo BSA-HTA em uma rede com as especificações de (LANGENDOEN; REIJERS, 2003)

É digno de nota que a ETAPA-IV, de refinamento, dos algoritmos propostos nesta dissertação tende a apresentar um maior custo de comunicação que a etapa de refinamento dos algoritmos *Robust Positioning* e *N-hop Multilateration*. Durante o refinamento, o mé-

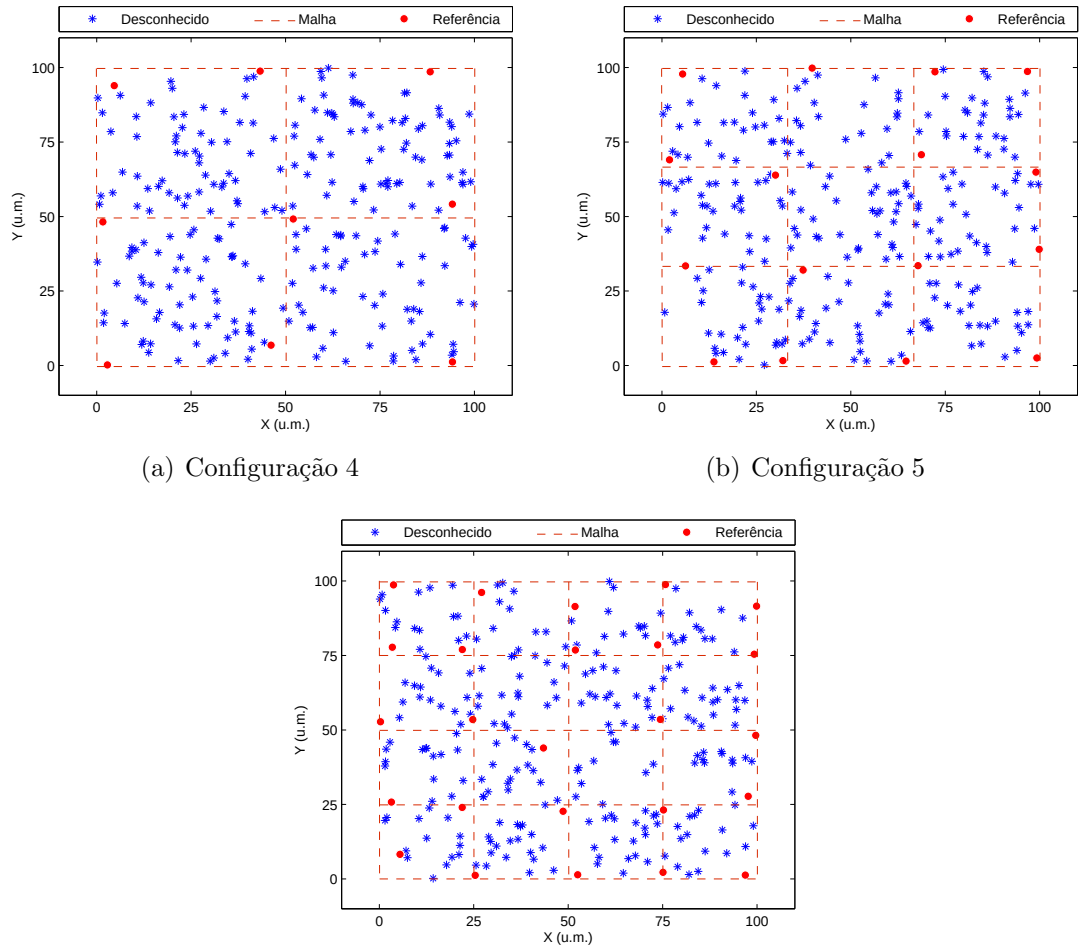
todo da *Lateração*, empregado nos algoritmos *Robust Positioning* e *N-hop Multilateration*, necessita apenas da informação das coordenadas dos nós vizinhos a um salto de distância. Já a ETAPA-IV dos algoritmos propostos nesta dissertação necessitam da informação das coordenadas e do fator de confiança dos nós vizinhos a um e dois saltos de distância, o que, numa análise qualitativa, sugere um maior tráfego de mensagens nos algoritmos ora propostos.

5.1.2 Avaliação Quanto ao Número de Nós de Referência e Desconhecidos

Nesta seção, os quatro algoritmos propostos são avaliados quanto ao número de nós de referência e de nós desconhecidos na rede. Assim como na Seção 5.1.1, os cenários foram gerados em uma área quadrada $A = 100 \times 100$ u.m., tendo como base as regras especificadas por (LANGENDOEN; REIJERS, 2003). Entretanto, nestes cenários, utilizou-se combinações de 150, 250 e 350 nós desconhecidos com 9, 16 e 25 nós de referência. Para a seleção dos 9, 16 e 25 nós de referência, foram utilizadas malhas quadradas de 3×3, 4×4 e 5×5 vértices, respectivamente. Adotou-se a distância limite de medição $L = 14$ u.m. Cabe lembrar que os resultados apresentados nesta seção não foram influenciados pelo MCMM, tendo em vista que, nestas simulações, todos os nós desconhecidos utilizaram mais do que dois nós de referência para estimar sua posição inicial, não atendendo à condição *(i)* do Algoritmo 5, definida na Seção 4.2.3. A Tabela 2 apresenta um resumo das configurações utilizadas para a geração dos referidos cenários. A Figura 19 apresenta o exemplo de três cenários com diferentes quantidades de nós de referência.

Tabela 2: Configurações dos cenários

Configuração	Número de nós			Vértices da malha
	Desconhecidos	Referências	Total	
1	150	9	159	3×3
2	150	16	166	4×4
3	150	25	175	5×5
4	250	9	259	3×3
5	250	16	266	4×4
6	250	25	275	5×5
7	350	9	359	3×3
8	350	16	366	4×4
9	350	25	375	5×5



(a) Configuração 4

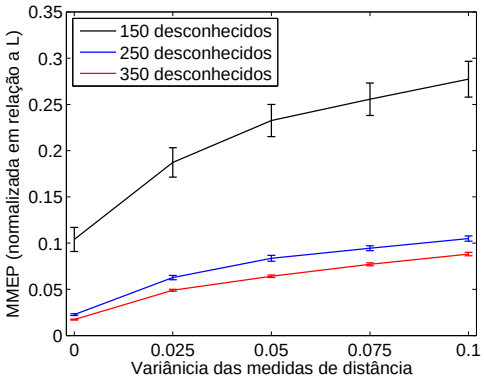
(b) Configuração 5

(c) Configuração 6

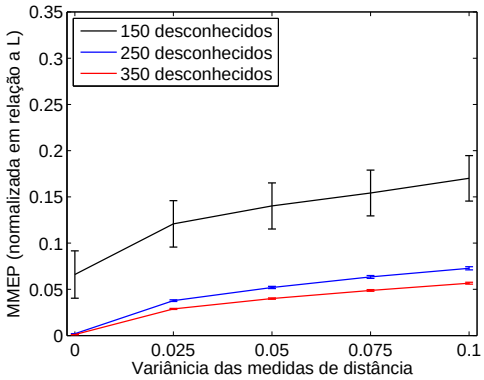
Figura 19: Exemplos de cenários com diferentes quantidades de nós de referência

Para cada configuração da Tabela 2, foram gerados 10 cenários distintos e calculada a $MMEP$ obtida pelos quatro algoritmos propostos. Os referidos valores de $MMEP$ constam no Apêndice A. As Figuras 20, 21, 22 e 23 apresentam os resultados alcançados por cada algoritmo, em cada uma das referidas configurações. Os resultados são expressos pela $MMEP$, normalizada em relação a L, em função da variância das medidas de distância. As $MMEPs$ normalizadas das Figuras 20, 21, 22 e 23 são apresentadas com os seus Intervalos de Confiança de 95%.

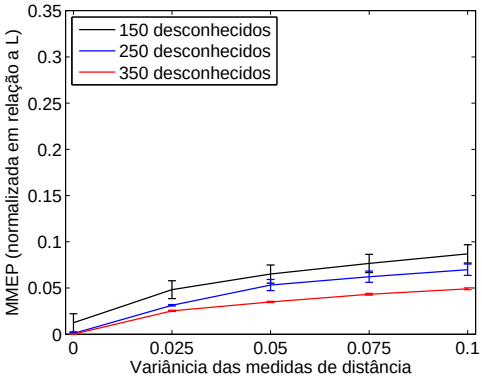
É possível constatar, pelas Figuras 20, 21, 22 e 23, que a $MMEP$ normalizada dos quatro algoritmos propostos tende a diminuir com o aumento do número de nós de referência da rede. Da mesma forma, o aumento do número de nós desconhecidos e, portanto, da conectividade da rede, reduz a $MMEP$ dos quatro algoritmos propostos, contribuindo positivamente com o desempenho dos algoritmos.



(a) 9 nós de referência

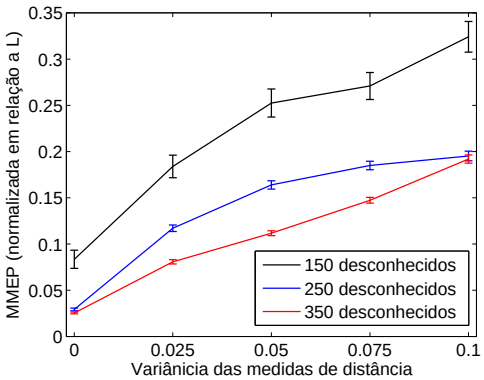


(b) 16 nós de referência

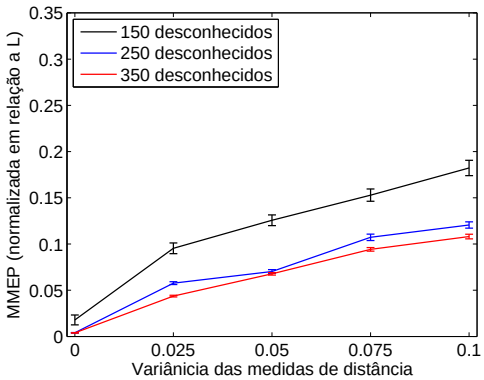


(c) 25 nós de referência

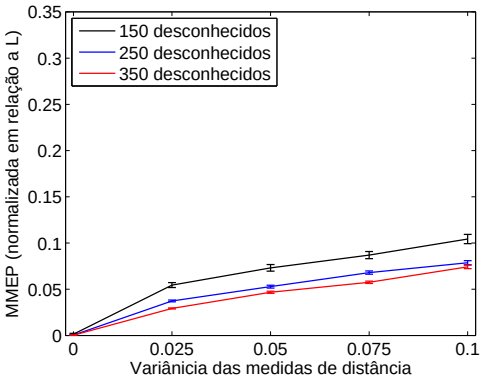
Figura 20: $MMEP$ do BSA-HTA quanto ao número de referências e nós desconhecidos



(a) 9 nós de referência

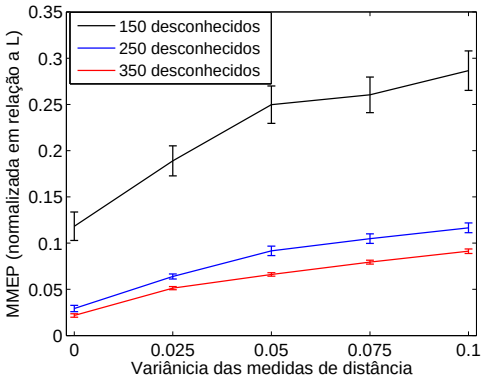


(b) 16 nós de referência

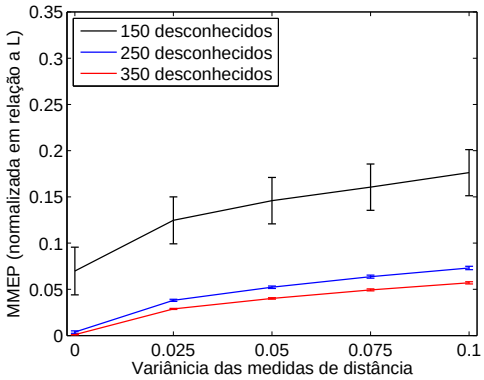


(c) 25 nós de referência

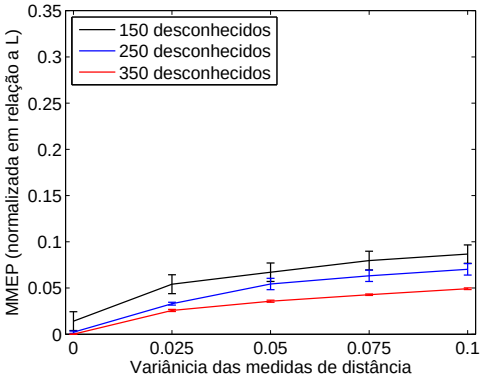
Figura 21: $MMEP$ do BSA-MMA quanto ao número de referências e nós desconhecidos



(a) 9 nós de referência

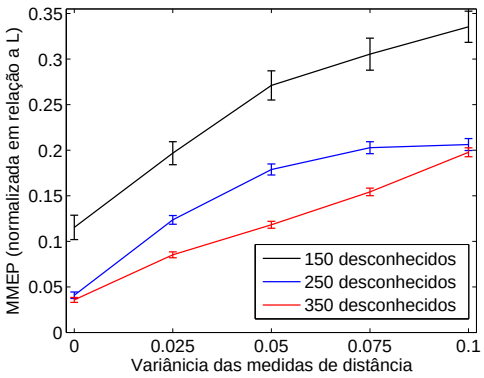


(b) 16 nós de referência

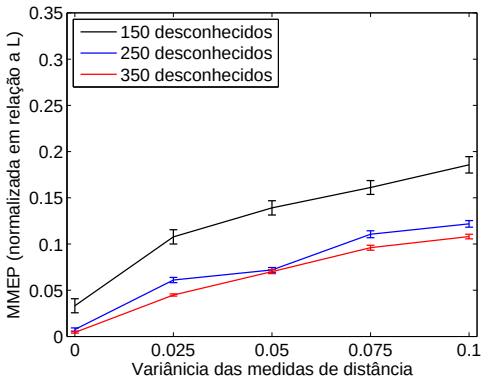


(c) 25 nós de referência

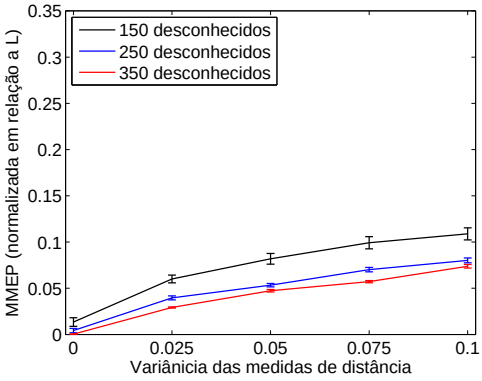
Figura 22: $MMEP$ do PSO-HTA quanto ao número de referências e nós desconhecidos



(a) 9 nós de referência



(b) 16 nós de referência



(c) 25 nós de referência

Figura 23: $MMEP$ do PSO-MMA quanto ao número de referências e nós desconhecidos

Considerando todas as simulações sem erro de medida de distância, o algoritmo BSA-HTA foi o que obteve o melhor desempenho, com uma $MMEP$ normalizada de $2,14 \times 10^{-5}$ u.m., *i.e.* um erro médio de posição de 3×10^{-4} u.m. em uma área de 100×100 u.m., em redes com 25 nós de referência e 350 nós desconhecidos. Já o algoritmo PSO-HTA foi o que obteve o pior desempenho, com uma $MMEP$ normalizada de $1,18 \times 10^{-1}$ u.m., *i.e.* um erro médio de posição de $1,66$ u.m. em uma área de 100×100 u.m., em redes com 150 nós desconhecidos e 9 nós de referência.

Nos casos com maior erro de medida de distância, *i.e.* com variância de medida de distância de 0,1, o algoritmo BSA-HTA foi o que obteve o melhor desempenho, com uma $MMEP$ normalizada de $4,917 \times 10^{-2}$ u.m., *i.e.* um erro médio de posição de $6,89 \times 10^{-1}$ u.m. em uma área de 100×100 u.m., em redes com 25 nós de referência e 350 nós desconhecidos. Já o algoritmo PSO-MMA foi o que obteve o pior desempenho, com uma $MMEP$ normalizada de $3,35 \times 10^{-1}$ u.m., *i.e.* um erro médio de posição de $4,01$ u.m. em uma área de 100×100 u.m., em redes com 150 nós desconhecidos e 9 nós de referência.

A Tabela 3 apresenta os algoritmos que obtiveram a menor $MMEP$ em cada configuração de rede, para cada variância das medidas de distância. Analisando a referida tabela, podemos constatar que o algoritmo BSA obteve melhores resultados que o PSO, exceto em três casos: nas configurações de rede 3, 8 e 9 com as variâncias de medida de distância de 0,1, 0 e 0,075, respectivamente. No que diz respeito ao fator de confiança, o HTA-CF obteve, na maioria dos casos, um melhor desempenho que o MMA-CF. Entretanto, em alguns casos, o MMA-CF obteve melhor desempenho. Conforme a Tabela 3, o melhor desempenho do MMA-CF tem maior incidência em redes com menos nós desconhecidos, *i.e.* nas configurações 1, 2 e 3, cujo número de nós desconhecidos é 150, e com baixo erro de medição de distância. Com a aumento do erro das medidas de distância, as áreas S_i, estimadas pelo Min-Max, se tornam mais imprecisas, degradando a qualidade do fator de confiança MMA-CF. Em contrapartida, o aumento do erro das medidas de distância não impacta o HTA-CF, uma vez que este computa apenas o número de saltos para os nós de referência.

Uma avaliação quanto ao tempo de processamento consumido na ETAPA-IV dos algoritmos propostos, tanto com o PSO quanto com o BSA, é apresentada na Figura 24, onde é possível relacionar o crescimento deste tempo com o aumento da conectividade da rede. Os tempos de processamento apresentados são normalizados em relação ao maior

Tabela 3: Algoritmos que obtiveram melhor desempenho

Configuração	Variância das medidas de distância				
	0,000	0,025	0,050	0,075	0,100
1	BSA-MMA	BSA-MMA	BSA-HTA	BSA-HTA	BSA-HTA
2	BSA-MMA	BSA-MMA	BSA-MMA	BSA-MMA	BSA-HTA
3	BSA-MMA	BSA-HTA	BSA-HTA	BSA-HTA	PSO-HTA
4	BSA-HTA	BSA-HTA	BSA-HTA	BSA-HTA	BSA-HTA
5	BSA-HTA	BSA-HTA	BSA-HTA	BSA-HTA	BSA-HTA
6	BSA-MMA	BSA-HTA	BSA-MMA	BSA-HTA	BSA-HTA
7	BSA-HTA	BSA-HTA	BSA-HTA	BSA-HTA	BSA-HTA
8	PSO-HTA	BSA-HTA	BSA-HTA	BSA-HTA	BSA-HTA
9	BSA-HTA	BSA-HTA	BSA-HTA	PSO-HTA	BSA-HTA

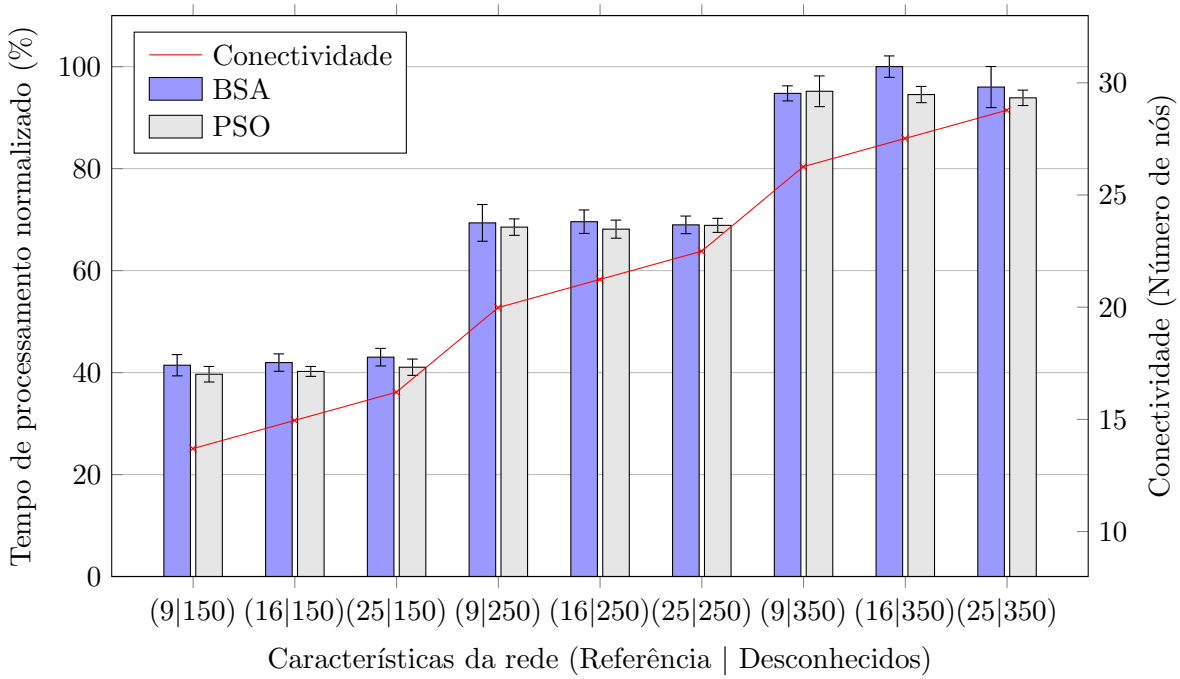


Figura 24: Tempos de processamento e conectividade da rede

tempo de processamento. Adicionalmente, na Figura 24, são apresentados os Intervalos de Confiança de 95% de cada tempo de processamento.

Para estimar a conectividade da rede consideramos, primeiramente, a Equação 31 que expressa a probabilidade Pr de um nó k ser vizinho de um nó desconhecido i:

$$Pr_{(k \text{ vizinho de } i)} = \frac{\pi L^2}{A}, \tag{31}$$

onde, k pode ser tanto um nó desconhecido como um nó de referência, L é a distância limite de medição e A é a área onde os nós da rede estão uniformemente distribuídos. Sendo assim, a conectividade de um nó desconhecido i corresponde à soma das probabilidades

de um nó k ser vizinho de i, para todo $k \neq i$, conforme a Equação 32:

$$conectividade = \sum_{k \neq i} Pr_{(k \ vizinho \ de \ i)}, \tag{32}$$

que pode ser reescrita conforme a Equação 33 (SÁ; NEDJAH; MOURELLE, 2014a, 2014b):

$$conectividade = \frac{(I + R - 1)\pi L^2}{A}, \tag{33}$$

relembrando que R é o total de nós de referência, I é o total de nós desconhecidos na rede e $\pi = 3,14$.

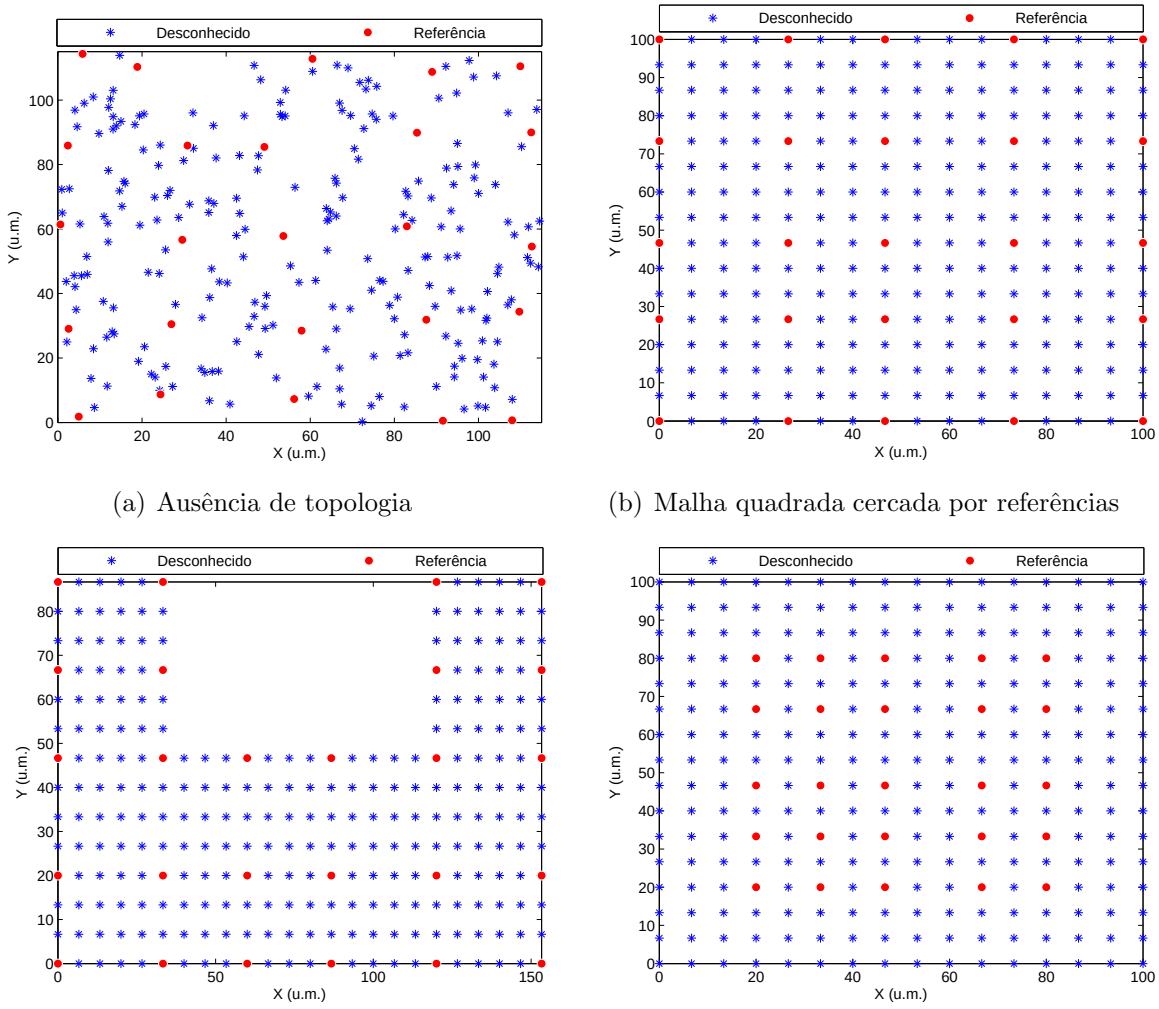
Constatamos que os tempos de processamento requeridos por ambos os algoritmos de otimização, PSO e BSA, são equivalentes, não excedendo uma diferença de $5,48\%$ em todos os casos. Cabe ressaltar que não houve diferença significativa no tempo de processamento no que diz respeito ao uso do HTA-CF ou do MMA-CF. Isto se deve ao fato de que ambos representam uma pequena parcela do esforço computacional demandado por todo o algoritmo.

5.1.3 Avaliação Quanto a Diferentes Topologias

Nesta seção, são apresentados os resultados obtidos pelos quatro algoritmos propostos em redes com diferentes topologias. Para isto, o número de nós desconhecidos da rede foi fixado em torno de 256 nós e o número de nós de referência foi fixado em torno 25 nós (10% da rede). As topologias avaliadas são apresentadas na Figura 25. A Figura 25a apresenta uma rede com ausência de topologia, gerada com base nas regras especificadas por (LANGENDOEN; REIJERS, 2003) e apresentadas na Seção 5.1.1, onde os nós de referência foram escolhidos com base em uma malha de 5×5 vértices. Para este tipo de rede, foram gerados 10 cenários distintos. A Figura 25b apresenta uma rede com topologia em malha, na forma de um quadrado. A Figura 25c apresenta uma rede com topologia em malha, na forma de um C. Para avaliar o desempenho do algoritmo em cenários onde há nós desconhecidos fora da área cercada pelos nós de referência, foram realizadas ainda simulações no cenário apresentado na Figura 25d. Nas topologias das Figuras 25a, 25b, e 25c, diferente do que ocorre na topologia da Figura 25d, é possível observar que todos os nós desconhecidos estão contidos em uma área cercada por nós de referência. O caso da topologia da Figura 25d constitui um desafio pois, segundo (LANGENDOEN; REIJERS, 2003), a maioria dos algoritmos de localização são muito sensíveis à presença, ou ausência,

de referências nas extremidades da rede, e algoritmos como o de *Lateração* possuem melhor desempenho quando as referências cercam os nós desconhecidos. Do mesmo modo, com esta topologia, objetiva-se avaliar a capacidade dos algoritmos propostos em transpor a limitação imposta na ETAPA-II pelo Min-Max onde, segundo (SAVVIDES; PARK; SRIVASTAVA, 2002), as posições estimadas estarão sempre contidas na área cercada por nós de referência.

Em todas as redes, adotou-se a distância limite de medição $L = 14$ u.m. As áreas ocupadas pelas redes foram dimensionadas de forma a proporcionar, aproximadamente, a mesma conectividade nas quatro topologias. A quantidade exata de nós utilizados em cada rede, bem como as dimensões das áreas onde as redes foram alocadas, são apresentadas na Tabela 4.



(a) Ausência de topologia

(b) Malha quadrada cercada por referências

(c) Malha na forma de um C

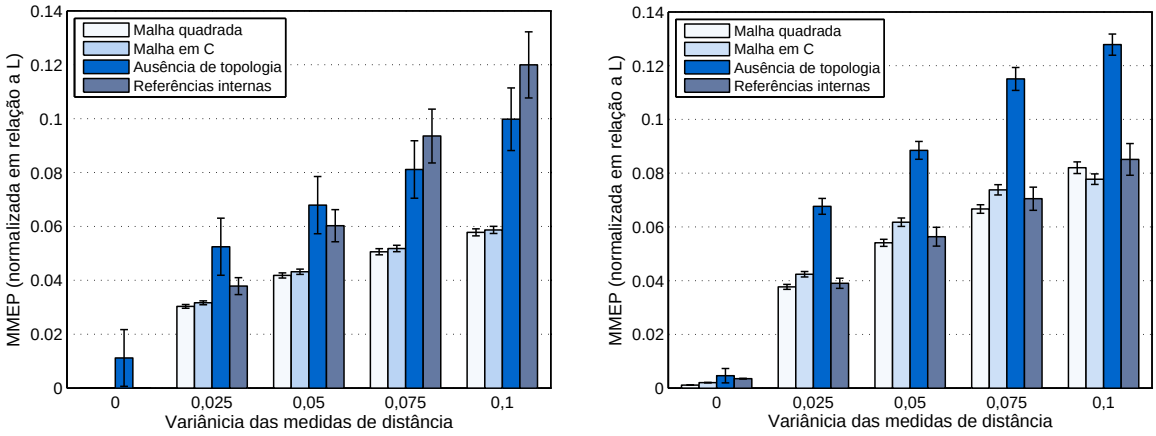
(d) Malha quadrada com referências internas

Figura 25: Diferentes topologias avaliadas

Tabela 4: Configurações das topologias

Figura	Número de nós			Área ocupada
	Desconhecidos	Referências	Total	
25a	231	25	256	115×115 u.m.
25b	231	25	256	100×100 u.m.
25c	239	26	264	$153,33 \times 86,67$ u.m.
25d	231	25	256	100×100 u.m.

A Figura 26 apresenta as $MMEP$ normalizadas alcançadas por cada algoritmo, quando aplicados às diferentes topologias, para diferentes variâncias de medida de distância. Os IC das $MMEP$ apresentadas na Figura 26 são de 95%. Os valores das $MMEP$ constam no Apêndice A.



(a) BSA-HTA (b) BSA-MMA

(c) PSO-HTA (d) PSO-MMA

Figura 26: Influência da topologia no desempenho dos quatro algoritmos

De uma forma geral, podemos observar que os melhores resultados foram obtidos na topologia em malha quadrada. A topologia em malha na forma de C apresentou $MMEPs$ equivalentes, entretanto, ligeiramente maiores do que na topologia em malha

quadrada. Atribuímos esta redução de desempenho à diferença de conectividade média das redes, causada pelo maior número de nós desconhecidos no perímetro da topologia de malha em C (cerca de 41% a mais do que na malha quadrada). Os nós desconhecidos que residem sobre o perímetro das topologias em malha possuem menor conectividade do que os demais nós da rede. Na topologia de malha quadrada, cerca de 19% dos nós desconhecidos da rede estão no perímetro da topologia, causando uma conectividade média de 10,96 nós. Já na topologia de malha em C, cerca de 26% dos nós desconhecidos estão no perímetro da rede, o que resulta numa conectividade média de 10,6 nós, *i.e.* menor do que na malha quadrada.

Nas redes com ausência de topologia, as $MMEPs$ alcançadas foram maiores do que nas redes em malha quadrada e em C. Ocorre que, ao serem aleatoriamente distribuídos, alguns nós desconhecidos podem ficar com a conectividade reduzida, mesmo que as redes tenham conectividades médias aproximadamente iguais. A título de exemplo, os nós destacados na Figura 27 possuem conectividades de 4 e 5 nós, ao passo que a menor conectividade nas redes em malha quadrada e em C é de 7 nós. A conectividade média da rede em questão é de 10,99 nós. Na Figura 27, é possível constatar que as posições dos referidos nós não foram corretamente estimadas, contribuindo para o aumento da $MMEP$.

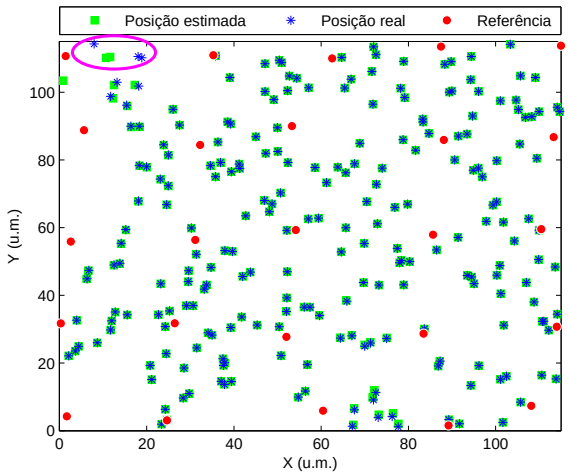
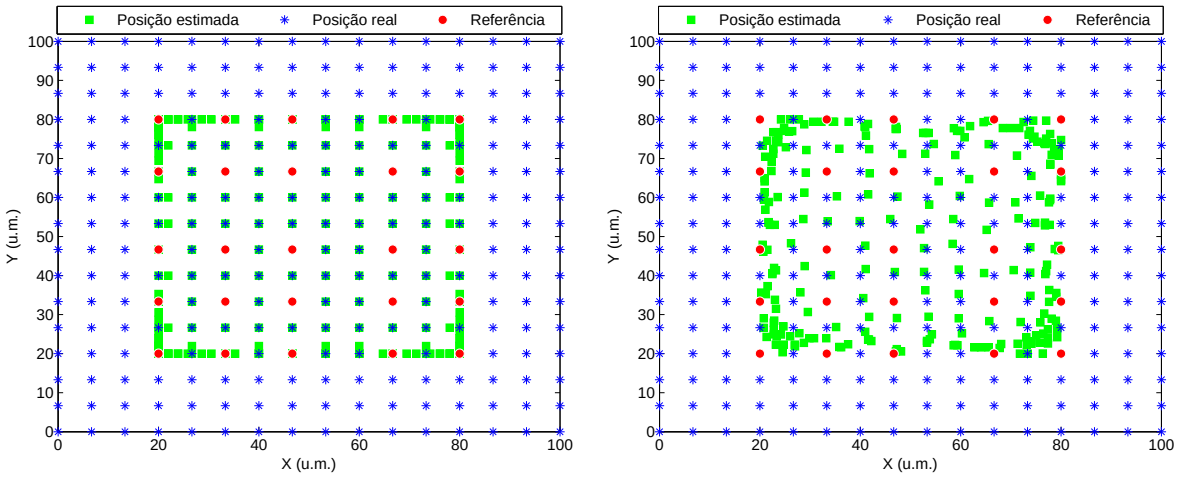


Figura 27: Exemplo de erro causado por uma baixa conectividade

No que tange às redes com referências internas, é possível constatar na Figura 26 um aumento significativo das $MMEPs$ nos casos em que foram utilizados os algoritmos baseados no PSO. Com o algoritmo BSA-HTA, o aumento da $MMEP$ não foi tão significativo, sendo mais expressivo nos casos com maior variância nas medidas de distância. Já

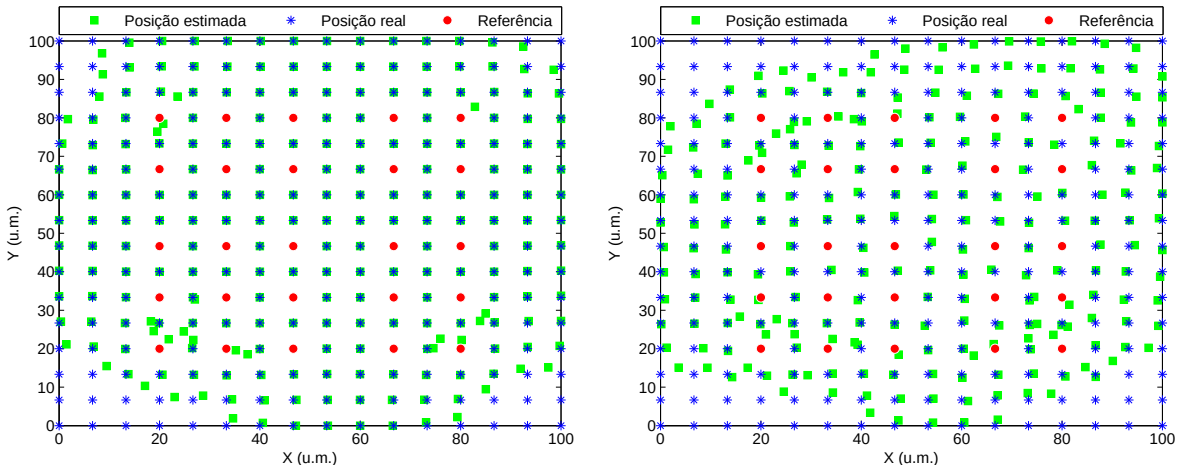
no caso do BSA-MMA, o fato de haver nós desconhecidos fora da área delimitada pelas referências não impactou significativamente no resultado. Neste caso, é possível observar que as $MMEPs$ alcançadas se aproximam das $MMEP$ obtidas no caso da topologia em malha quadrada, denotando uma boa capacidade do BSA-MMA em localizar nós que não estejam cercados por referências.

A Figura 28 apresenta exemplos de resultados obtidos pelo Min-Max, ao fim da ETAPA-II, em redes com referências internas e para diferentes variâncias de medida de distância. As Figuras 29, 30, 31 e 32 apresentam exemplos de resultados obtidos ao final da ETAPA-IV dos algoritmos PSO-HTA, PSO-MMA, BSA-HTA e BSA-MMA, respectivamente, considerando as mesmas variâncias de medida de distância. Constatamos que os quatro algoritmos foram capazes de transpor a limitação imposta na ETAPA-II pelo Min-Max onde, segundo (SAVVIDES; PARK; SRIVASTAVA, 2002), as posições estimadas estarão sempre contidas na área cercada por nós de referência. Verificamos que, ao fim da ETAPA-IV, nos quatro algoritmos, os nós mais afastados da área onde estão contidas as referências tendem a ter um maior erro de posição. Constatamos ainda que, com o aumento do erro das medidas de distância, o erro de posição destes nós mais afastados tende a aumentar numa proporção maior do que ocorre com os nós que estão cercados por referências. Este comportamento, entretanto, é mitigado pelo BSA-MMA, conforme exemplificado na Figura 32, onde o erro de posição dos nós mais afastados das referências é atenuado.



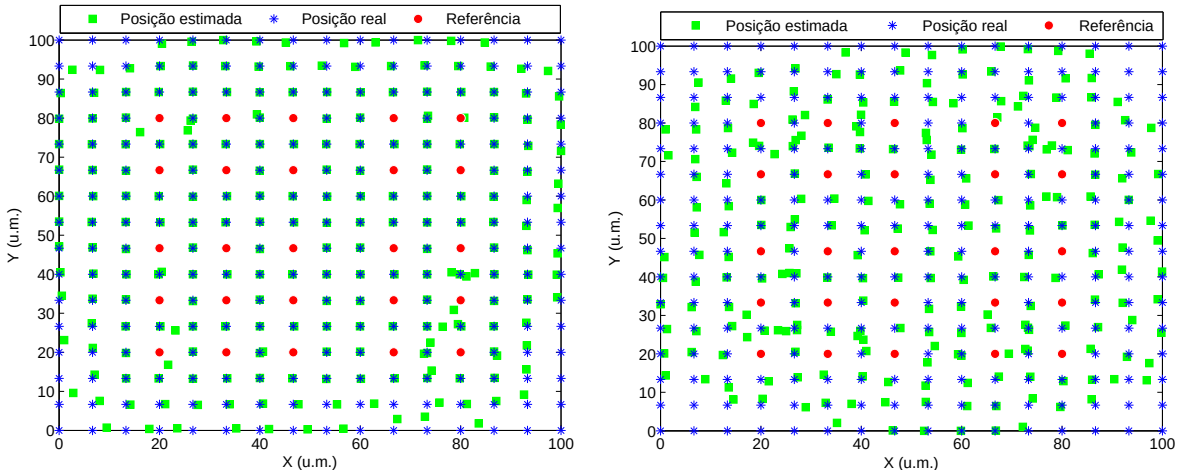
(a) Variância de medida de distância 0 (b) Variância de medida de distância 0,1

Figura 28: Resultados obtidos pelo Min-Max em redes com referências internas



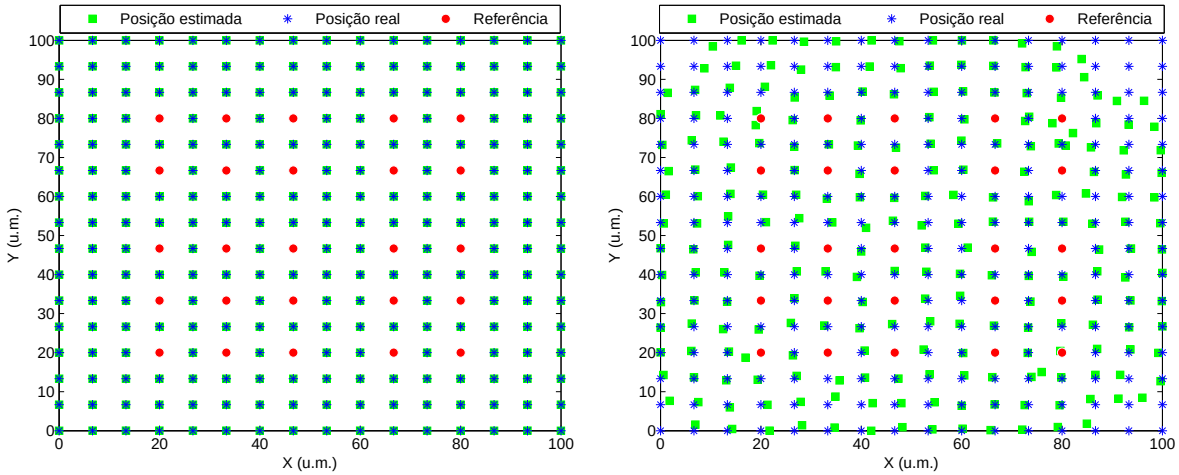
(a) Variância de medida de distância 0 (b) Variância de medida de distância 0,1

Figura 29: Resultados obtidos pelo PSO-HTA em redes com referências internas



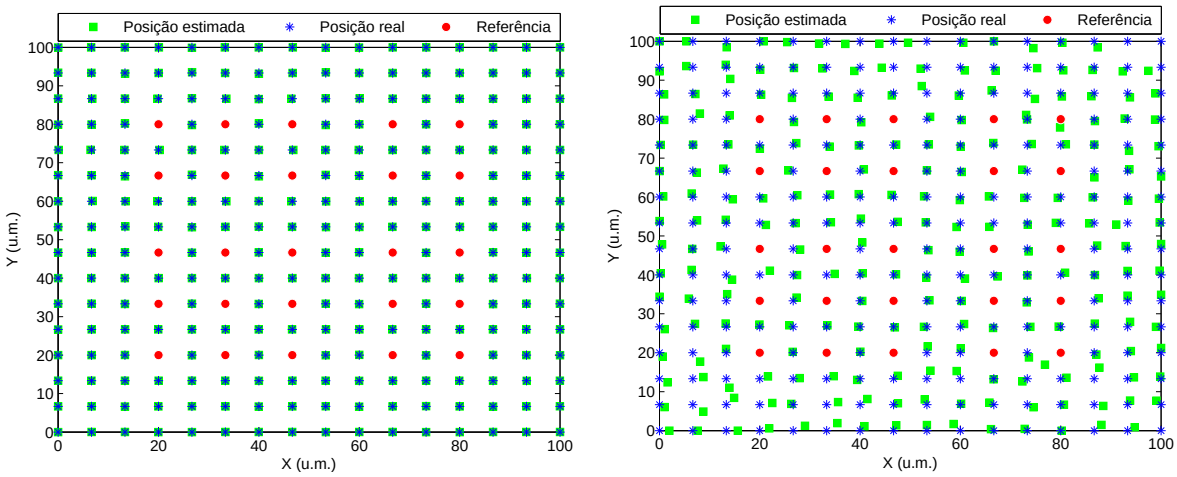
(a) Variância de medida de distância 0 (b) Variância de medida de distância 0,1

Figura 30: Resultados obtidos pelo PSO-MMA em redes com referências internas



(a) Variância de medida de distância 0 (b) Variância de medida de distância 0,1

Figura 31: Resultados obtidos pelo BSA-HTA em redes com referências internas



(a) Variância de medida de distância 0

(b) Variância de medida de distância 0,1

Figura 32: Resultados obtidos pelo BSA-MMA em redes com referências internas

A Tabela 5 apresenta os algoritmos que obtiveram a melhor $MMEP$ em cada tipo de topologia, para cada variância das medidas de distância. Podemos constatar que o algoritmo BSA obteve melhores resultados que o PSO, exceto em quatro situações. Entretanto, mesmo nas situações em que o PSO, especificamente o PSO-HTA, obteve melhores resultados, o desempenho do BSA-HTA se mostrou equivalente, apresentando menos de 1% de diferença entre as $MMEPs$ normalizadas. No que diz respeito ao fator de confiança, o HTA-CF obteve, na maioria dos casos, um melhor desempenho que o MMA-CF. Entretanto, em alguns casos, o MMA-CF obteve melhor desempenho. Conforme a Tabela 5, o MMA-CF apresentou melhor desempenho na topologia com referências internas e variância de medição de distância maior ou igual a $0,5$, bem como nas simulações com ausência de topologia e sem erro de medição de distância.

Tabela 5: Algoritmos que obtiveram melhor desempenho em cada tipo de topologia

	Variância das medidas de distância				
Topologia	0,000	0,025	0,050	0,075	0,100
Malha quadrada	BSA-HTA	PSO-HTA[1]	PSO-HTA[1]	BSA-HTA[1]	BSA-HTA[1]
Malha em C	BSA-HTA	BSA-HTA[1]	PSO-HTA[1]	BSA-HTA[1]	BSA-HTA[1]
Ausência de topologia	BSA-MMA	BSA-HTA[1]	PSO-HTA[1]	BSA-HTA[1]	BSA-HTA[1]
Referências internas	BSA-HTA	BSA-HTA	BSA-MMA	BSA-MMA	BSA-MMA

[1] Simulações em que o desempenho dos algoritmos BSA-HTA e PSO-HTA se mostraram equivalentes, apresentando menos de 1% de diferença entre as $MMEPs$ normalizadas.

5.1.4 Avaliação do Método MCMM

Para avaliar o desempenho do método MCMM, foram gerados cenários com base nas mesmas regras especificadas por (LANGENDOEN; REIJERS, 2003) e descritas na Seção 5.1.1. Primeiramente, foram gerados cenários com 150, 250 e 350 nós desconhecidos e 4 nós de referência. A seleção dos 4 nós de referência foi feita utilizando uma malha quadrada de 2×2 vértices. Os nós foram distribuídos em uma área de simulação $A = 100 \times 100$ u.m. e a distância limite de medição adotada foi $L = 14$ u.m. Para simular restrições quanto à divulgação dos nós de referência, foi imposto o máximo de 6 saltos para retransmissão de mensagens da ETAPA-I, fazendo com que os nós desconhecidos não fossem capazes de reconhecer referências além deste limite. A Figura 33 apresenta um exemplo de cenário com 4 de nós de referência e 250 nós desconhecidos.

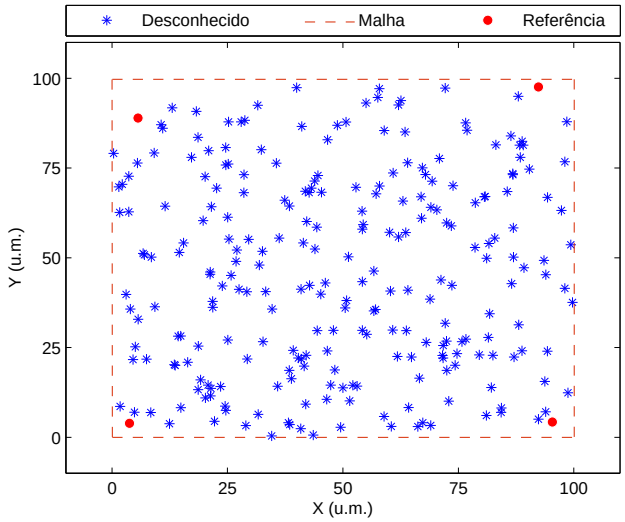
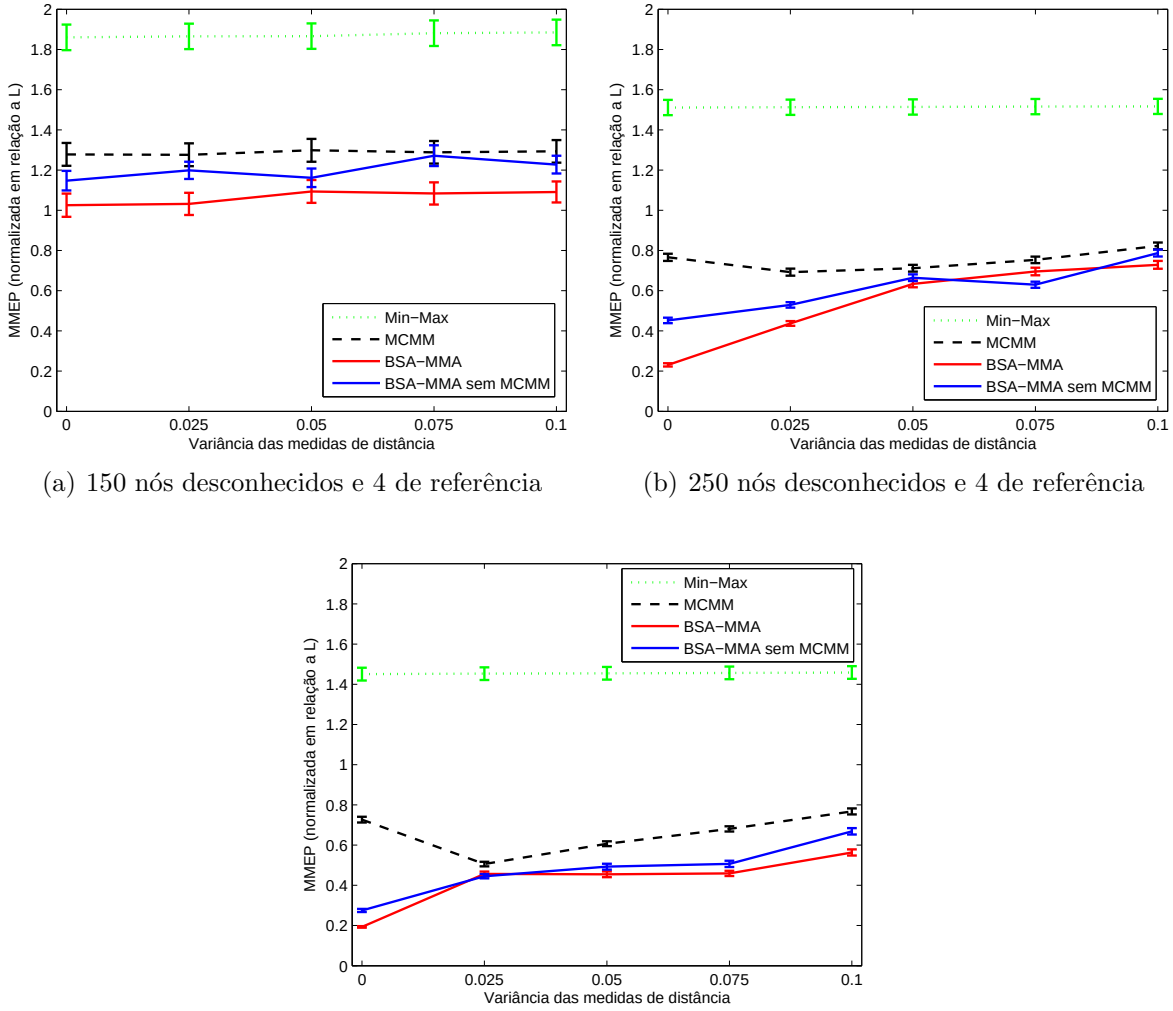


Figura 33: Exemplo de cenário com 4 nós de referência e 250 nós desconhecidos

Para cada quantidade de nós desconhecidos, foram gerados 10 cenários distintos e calculada a $MMEP$ obtida ao fim das ETAPAS-II, III e IV do algoritmo de localização. Para simplificar a análise, utilizou-se apenas a versão BSA-MMA, tendo em vista que os resultados obtidos pelo Min-Max e pelo MCMM independem do algoritmo de otimização e do fator de confiança empregados na ETAPA-IV. A Figura 34 apresenta as $MMEP$ normalizadas obtidas pelos algoritmos Min-Max, MCMM e BSA, nas ETAPAS-II, III e IV, respectivamente, em redes com 150, 250 e 350 nós desconhecidos e 4 nós de referência. É apresentada ainda, na Figura 34, a $MMEP$ obtida ao fim da ETAPA-IV do BSA-

MMA, quando aplicado sem a execução do MCMM na ETAPA-III. Os IC das $MMEP$ apresentadas na Figura 34 são de 95%.



(a) 150 nós desconhecidos e 4 de referência

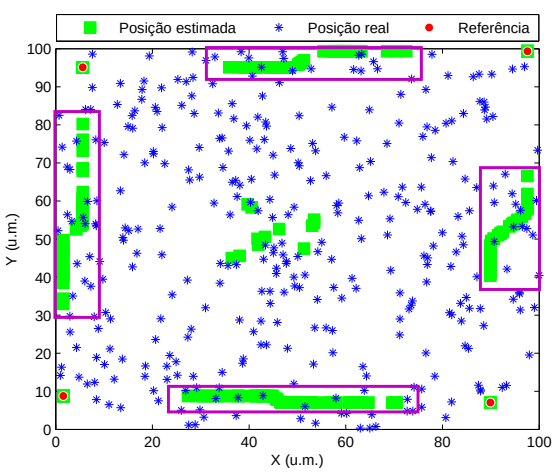
(b) 250 nós desconhecidos e 4 de referência

(c) 350 nós desconhecidos e 4 de referência

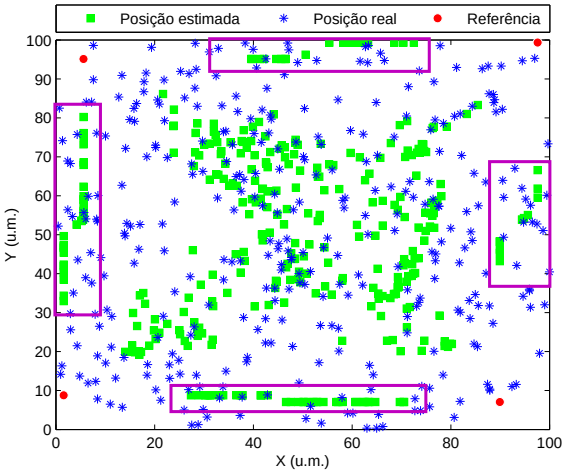
Figura 34: $MMEP$ normalizadas obtidas pelos algoritmos Min-Max, MCMM e BSA.

Na Figura 34, é possível verificar o ganho obtido pelo método MCMM em situações onde a qualidade das posições iniciais estimadas pelo Min-Max foram degradadas pelo fato de alguns nós da rede não terem conhecido mais do que dois nós de referência durante a ETAPA-I. Em redes com 150 nós desconhecidos, o MCMM foi capaz de reduzir 30% a 32% da $MMEP$ normalizada obtida pelo Min-Max. Em redes com 250 nós desconhecidos, tal redução foi de 46% a 54%. Já em redes com 350 nós desconhecidos a redução foi de 47% a 65%. No que se refere ao resultado final do processo de localização, constatamos que, de uma forma geral, os resultados obtidos pelo BSA-MMA, com a execução do MCMM, foram melhores do que os resultados obtidos pelo BSA-MMA sem a execução do MCMM.

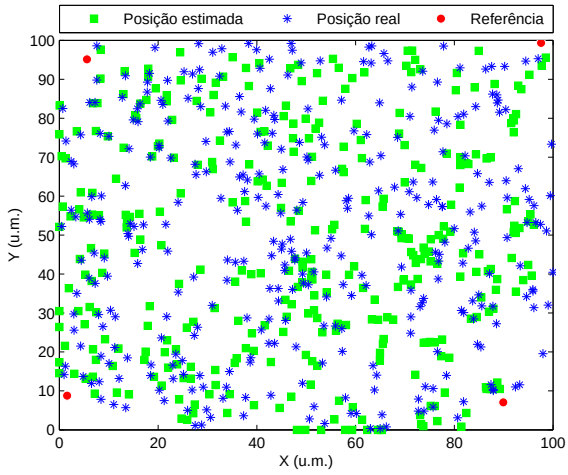
A título de exemplo, a Figura 35 apresenta o resultado de cada etapa do processo de localização em uma rede com 350 nós desconhecidos, 4 nós de referência e variância de medida de distância de 0,1.



(a) ETAPA-II (Min-Max)



(b) ETAPA-III (MCMM)



(c) ETAPA-IV (BSA-MMA)

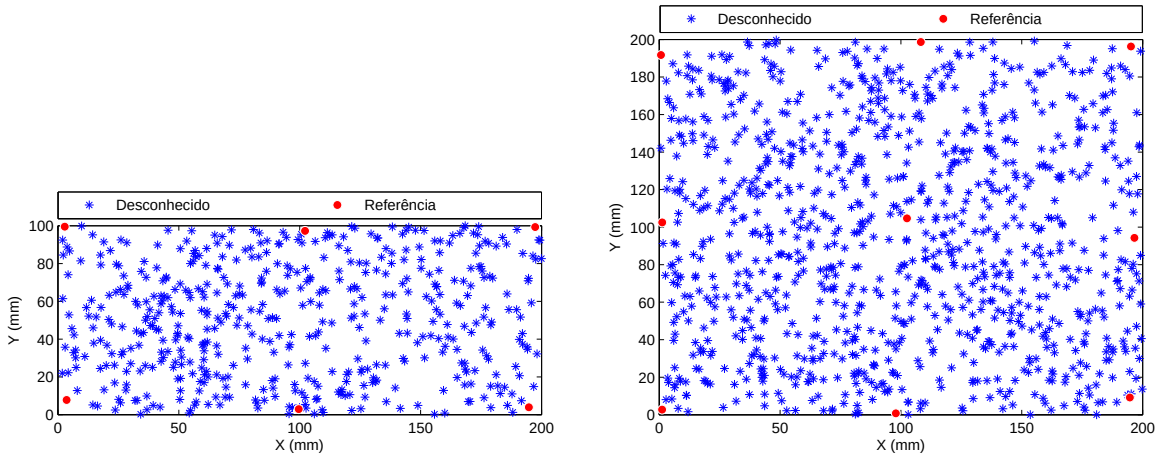
Figura 35: Evolução do processo de localização nas ETAPAS-II, III e IV

Na Figura 35a, podemos observar que, após a execução do Min-Max, a maioria dos nós desconhecidos não foi capaz de estimar adequadamente sua posição, devido à restrição na divulgação dos nós de referência, resultando em uma alta $MMEP$ ao fim da ETAPA-II. A parcela de nós desconhecidos que se baseou em apenas duas referências teve sua posição estimada dentro de uma das áreas destacadas, entre as duas respectivas referências. A parcela de nós desconhecidos que se baseou em apenas um nó de referência estimou sua posição como sendo a mesma da referência tomada como base.

Na Figura 35b, é possível constatar que, após a execução do MCMM, os nós per-

tencentes às duas parcelas supramencionadas reavaliaram suas posições com base nos demais nós desconhecidos da rede, causando uma melhora da $MMEP$. De acordo com o Algoritmo 5, os nós com alinhamento inferior ao limiar estabelecido de $\beta = 0,006$ não modificaram suas estimativas de posição, conforme destacado na Figura 35b, uma vez que estas posições já estariam relativamente próximas das suas posições reais. O resultado final após o refinamento da ETAPA-IV é apresentado na Figura 35c, demonstrando a capacidade do algoritmo em estimar a posição dos nós desconhecidos mesmo que, inicialmente, os mesmos encontrem dificuldades de adquirir informações dos nós de referência.

Adicionalmente, avaliou-se o desempenho do algoritmo em redes de diferentes tamanhos. Com base no cenário de 250 nós desconhecidos e 4 nós de referência, exemplificado na Figura 33, o qual será referido como célula, foram gerados mais duas configurações de rede. A primeira, apresentada na Figura 36a, corresponde a duas células dispostas lado a lado, formando um retângulo. A segunda, apresentada na Figura 36b, corresponde a quatro células dispostas juntas na forma de um quadrado.



(a) 2 células (b) 4 células

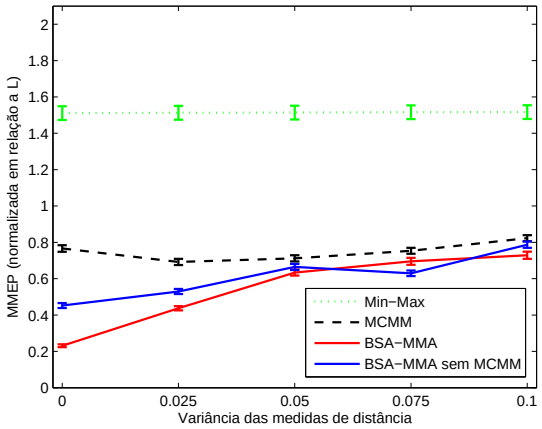
Figura 36: Exemplos de cenários de diferentes tamanhos, utilizados para avaliar o MCMM

A Tabela 6 apresenta as especificações das redes em função do número de células. Cabe ressaltar que não há diferença significativa na conectividade média das três configurações constantes na referida tabela, uma vez que estas possuem a mesma densidade de nós e que os mesmos foram posicionados aleatoriamente seguindo uma distribuição uniforme.

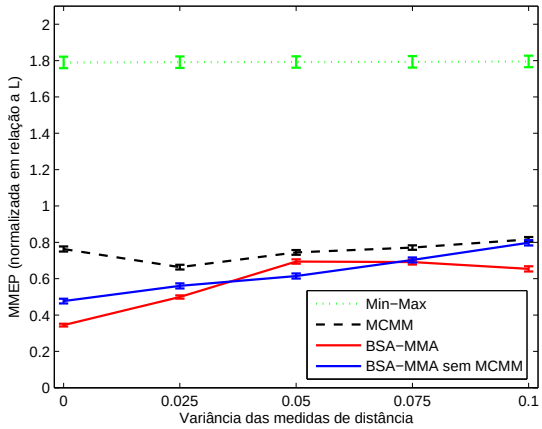
Tabela 6: Especificações das configurações de rede em função do número de células

Configuração	Células	Número de nós			Área
		Desconhecidos	Referências	Total	
(250|4)	1	250	4	254	$100u.m. \times 100u.m.$
(500|6)	2	500	6	506	$100u.m. \times 200u.m.$
(1000|9)	4	1000	9	1009	$200u.m. \times 200u.m.$

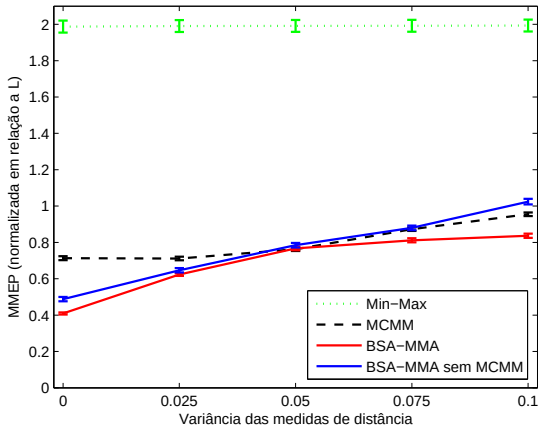
Para cada configuração de rede apresentada na Tabela 6, foram gerados 10 cenários distintos e calculada a $MMEP$ ao fim das ETAPAS-II, III e IV. Da mesma forma, nestas simulações, a retransmissão de mensagens da ETAPA-I foi limitada em 6 saltos, fazendo com que os nós desconhecidos não reconhecessem referências além deste limite. Os resultados obtidos são apresentados na Figura 37, para diferentes variâncias de medida de distância. Os IC das $MMEP$ apresentadas na Figura 37 são de 95%.



(a) 250 nós desconhecidos e 4 de referência



(b) 500 nós desconhecidos e 6 de referência



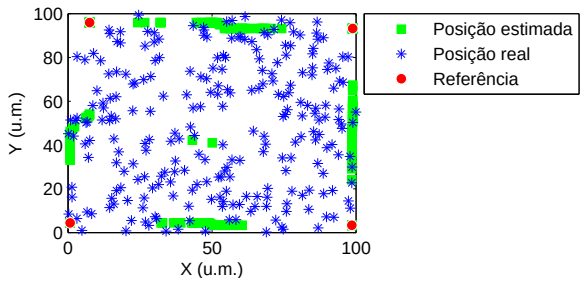
(c) 1000 nós desconhecidos e 9 de referência

Figura 37: Desempenho das ETAPAS-II, III e IV em redes de diferentes tamanhos

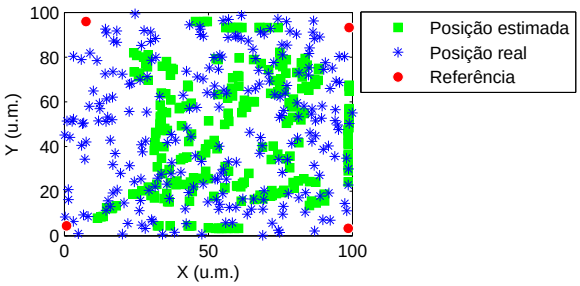
Na Figura 37, podemos constatar a contribuição do MCMM para a redução da $MMEP$ das redes avaliadas, em situações em que os nós desconhecidos tenham dificuldade de obter informações dos nós de referência. Na configuração (250|4), a $MMEP$ do MCMM foi 46% a 54% menor que a obtida pelo Min-Max. Na configuração (500|6), a $MMEP$ obtida pelo MCMM foi 55% a 63% menor que a do Min-Max. Já na configuração (1000|9), a $MMEP$ do MCMM foi 52% a 64% menor que a do Min-Max.

Após a execução do MCMM, as redes foram submetidas ao processo de refinamento da ETAPA-IV do BSA-MMA. No melhor caso, *i.e.* sem erro de medida de distância, a $MMEP$ normalizada final das configurações (250|4), (500|6) e (1000|9) foi de 0,23, 0,34 e 0,41, respectivamente. No pior caso, *i.e.* com variância de medida de distância de 0,1, a $MMEP$ normalizada final das configurações (250|4), (500|6) e (1000|9) foi de 0,73, 0,65 e 0,84, respectivamente. De uma forma geral, os resultados indicam que, quando há restrições na divulgação dos nós de referência, o desempenho do algoritmo é melhor em redes menores.

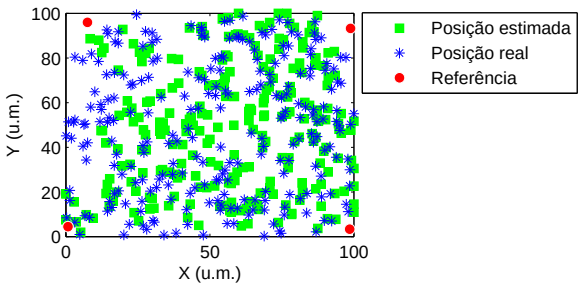
A evolução do processo de localização, apresentado nas Figuras 38, 39 e 40, demonstra a capacidade do algoritmo em estimar a posição dos nós desconhecidos mesmo que, inicialmente, os mesmos encontrem dificuldades de adquirir informações dos nós de referência.



(a) Min-Max

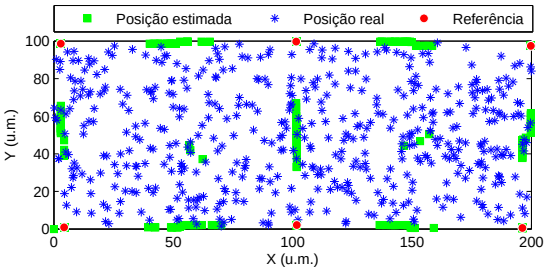


(b) MCMM

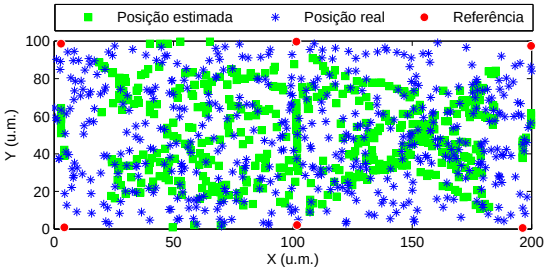


(c) BSA-MMA

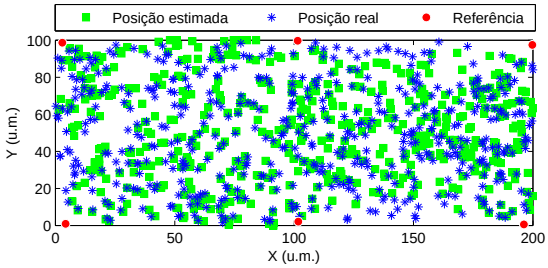
Figura 38: Localização em redes com 250 nós desconhecidos e 4 referências



(a) Min-Max

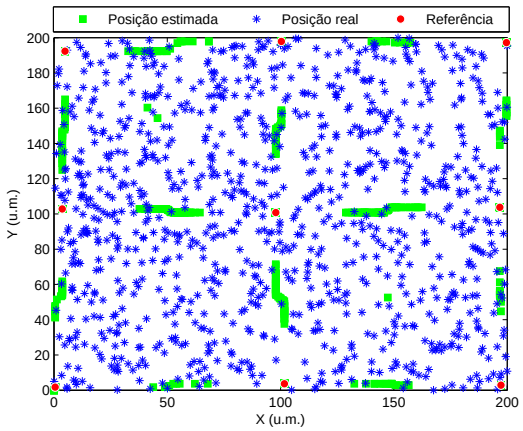


(b) MCMM

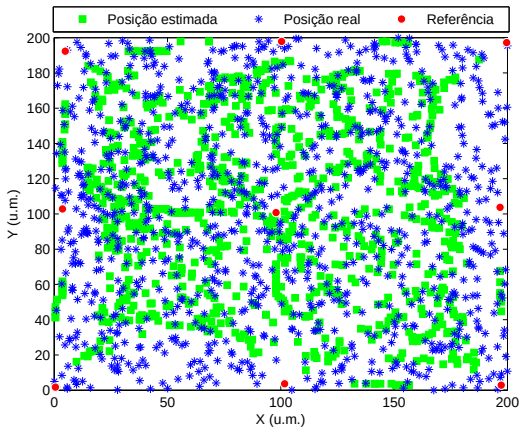


(c) BSA-MMA

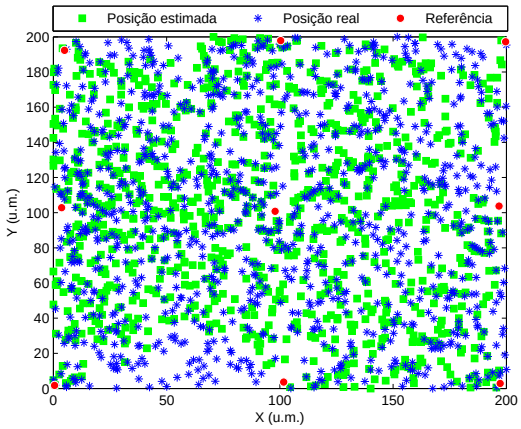
Figura 39: Localização em redes com 500 nós desconhecidos e 6 referências



(a) Min-Max



(b) MCMM



(c) BSA-MMA

Figura 40: Localização em redes com 1000 nós desconhecidos e 9 referências

5.2 Experimentos

Nesta seção, são apresentados os resultados de experimentos conduzidos em um enxame de robôs reais, do tipo *Kilobot*, cujas características são apresentadas na Seção 5.2.1.

Nas simulações apresentadas na Seção 5.1, os algoritmos baseados no BSA obtiveram, de uma forma geral, desempenhos melhores do que aqueles baseados no PSO. Por este motivo, foram escolhidos os algoritmos BSA-HTA e BSA-MMA para a execução dos experimentos desta seção. Os parâmetros utilizados na ETAPA-IV destes algoritmos foram os mesmos das simulações descritas na Seção 5.1, *i.e.* $K = 1$, $\Delta = 40$ e $\delta = 10$. O limiar de alinhamento do MCMM foi ajustado para $\beta = -0,05$.

Os resultados obtidos são apresentados nas Seções 5.2.2, 5.2.3 e 5.2.4. Primeiramente, na Seção 5.2.2, são apresentados os resultados de experimentos em enxames de diferentes tamanhos, considerando ainda diferentes proporções de nós de referência. Na Seção 5.2.3, são reportados os resultados obtidos em enxames com diferentes topologias. Cabe ressaltar que não houve influência do MCMM nos resultados apresentados nas Seções 5.2.2 e 5.2.3, tendo em vista que, nestes experimentos, todos os nós desconhecidos utilizaram mais do que dois nós de referência para estimar sua posição inicial, não atendendo a condição *(i)* do Algoritmo 5, definida na Seção 4.2.3. O desempenho do método MCMM é avaliado na Seção 5.2.4, onde são apresentadas situações em que os nós desconhecidos utilizam menos do que três nós de referência no cálculo da posição inicial da ETAPA-II.

5.2.1 O Robô

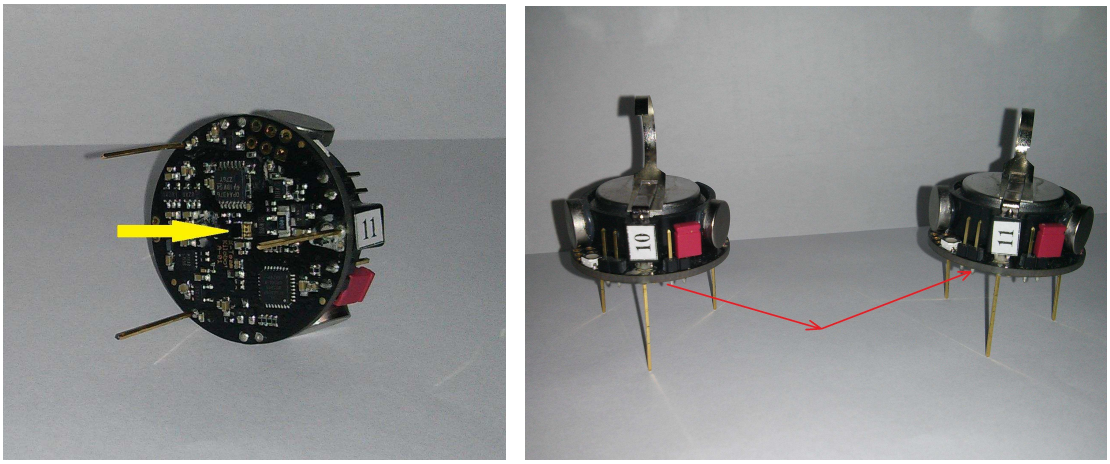
Para realizar os experimentos, foi utilizado um enxame de robôs do tipo *Kilobot* (RUBENSTEIN; AHLER; NAGPAL, 2012), apresentado na Figura 41. Os *Kilobots* são robôs de baixo custo, de $33mm$ de diâmetro, com habilidades necessárias para estabelecer comportamentos coletivos. Tais habilidades compreendem a capacidade de comunicação e medição de distância entre robôs vizinhos, processamento embarcado e locomoção.

Para se comunicar com robôs vizinhos, cada robô é dotado de um LED e de um fotodiodo infravermelho, para transmissão e recepção de mensagens, respectivamente. Conforme indicado na Figura 42a, ambos os dispositivos estão situados na parte inferior da placa de circuito impresso do robô, apontando para baixo, de forma que a mensagem transmitida aos outros robôs se propague por meio da reflexão na superfície em que os

(a) *Kilobot* (b) Enxame de *Kilobots*

Figura 41: Robôs do tipo *Kilobot*

mesmos se encontram. A propagação do sinal transmitido entre robôs é ilustrada na Figura 42b. Os diagramas de irradiação e de recepção dos sinais infravermelhos do robô são isotrópicos no plano horizontal, o que permite que o robô receba mensagens igualmente de todas as direções (RUBENSTEIN; AHLER; NAGPAL, 2012). O raio de comunicação do robô é de até $100mm$ e sua taxa de transmissão máxima é de $30kbps$ (RUBENSTEIN; AHLER; NAGPAL, 2012). Uma vez que os robôs compartilham o mesmo canal de comunicação, existe a possibilidade de dois ou mais robôs tentarem transmitir simultaneamente, causando colisões de mensagens. Para mitigar o problema de colisão, os robôs utilizam a técnica CSMA/CA (ZIOUVA; ANTONAKOPOULOS, 2002). Entretanto, conforme reportado em (RUBENSTEIN; AHLER; NAGPAL, 2012), mesmo com o uso desta técnica, em ambientes com muitos robôs, deve-se considerar a possibilidade de degradação da comunicação em função de eventuais colisões.



(a) LED e fotodiodo de comunicação (b) Propagação da mensagem

Figura 42: Comunicação entre *Kilobots*

Para medir distâncias o robô utiliza a técnica RSS, na qual a distância entre robôs é inferida pela medição da potência do sinal infravermelho recebido durante a troca de mensagens. A resolução das medidas de distância é de $1mm$. Note que, apesar de a resolução das medidas de distância dos robôs ser de $1mm$, as $MMEPs$ obtidas nos experimentos das Seções 5.2.2, 5.2.3 e 5.2.4 apresentam resoluções menores do que $1mm$. Devemos lembrar que o erro da posição estimada por um robô i corresponde à distância existente entre a posição calculada pelo algoritmo p_{calc} e a posição real do robô p_{real}, *i.e.* $||p_{real}(i) - p_{calc}(i)||$, conforme consta na Equação 29. Dessa forma, por exemplo, se $p_{real}(i) = (0mm, 0mm)$ e $p_{calc}(i) = (1mm, 1mm)$, então o erro da posição estimada pelo robô i será de aproximadamente $1,4142mm$, que necessita de uma resolução menor do que $1mm$ para ser representada. Consequentemente, as $MMEPs$ dos experimentos não serão necessariamente números inteiros e poderão ser representadas com resoluções menores do que $1mm$.

Cada mensagem transmitida pelo *Kilobot* comporta um total de $23bits$ de dados. Em nossa implementação, a mensagem foi formatada conforme a Figura 43, onde são alocados 7 bits para transmissão do número de identificação do robô transmissor, ou ID, 6 bits para compor um cabeçalho que identifica o tipo de informação que está sendo transmitida, e 10 bits para transmissão da informação desejada.

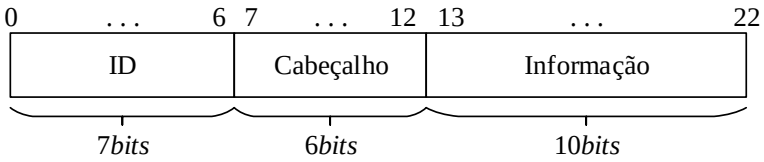


Figura 43: Formato de mensagem

No que diz respeito ao processamento embarcado, cada robô dispõe de um microprocessador ATMEL ATmega328p de $8MHz$ (ATMEL, 2015a). A linguagem de programação é C (KERNIGHAN; RITCHIE; EJEKLINT, 1988) e a compilação é feita por meio do compilador *WinAVR* (WINAVR, 2015), combinado com o *AVR Studio* (ATMEL, 2015b).

A capacidade de locomoção dos robôs não é explorada nesta dissertação, tendo em vista que os algoritmos propostos presumem que os nós estejam estáticos.

5.2.2 Avaliação Quanto ao Tamanho do Enxame

Os experimentos apresentados nesta seção foram conduzidos em enxames de 40, 54 e 66 robôs dispostos em uma topologia de malha retangular. Adicionalmente, para cada

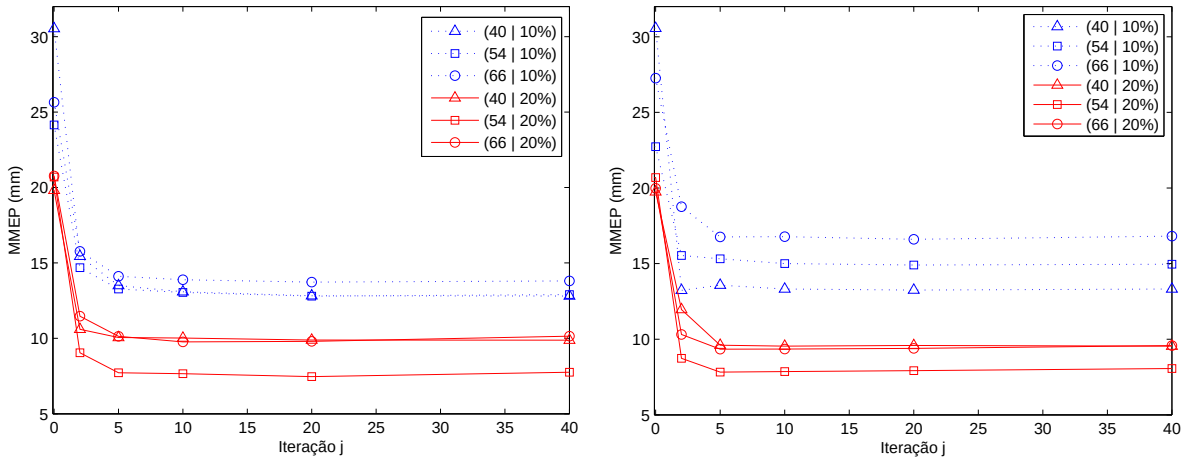
tamanho de enxame, foram realizados experimentos considerando aproximadamente 10% e 20% de nós de referência, resultando nas seis configurações apresentadas na Tabela 7. A distância limite de medição foi configurada como $L = 80mm$. Cabe ressaltar que, em todas as configurações de enxame consideradas nesta seção, o espaçamento entre robôs foi mantido o mesmo, de forma que não há diferenças significativas na conectividade de cada configuração. Os nós de referência foram dispostos conforme apresentado no Apêndice B.

Tabela 7: Configuração dos enxames

	Número de robôs			Percentual	
Configuração	Desconhecidos	Referências	Total	de referências	Área
(40|10%)	36	4	40	10,0%	$0,35m \times 0,20m$
(54|10%)	48	6	54	11,1%	$0,40m \times 0,25m$
(66|10%)	59	7	66	10,6%	$0,50m \times 0,25m$
(40|20%)	32	8	40	20,0%	$0,35m \times 0,20m$
(54|20%)	42	12	54	22,2%	$0,40m \times 0,25m$
(66|20%)	53	13	66	19,7%	$0,50m \times 0,25m$

Para cada configuração da Tabela 7, foram realizados quatro experimentos, dois com cada um dos algoritmos avaliados, BSA-HTA e BSA-MMA. A evolução do processo de localização pode ser observada na Figura 44, onde são apresentadas as $MMEPs$ (mm) obtidas ao longo das iterações j da ETAPA-IV do Algoritmo 7. As $MMEPs$ apresentadas na iteração 0 correspondem aos resultados obtidos pelo Min-Max ao término da ETAPA-II. Lembramos que, nestes experimentos, as posições iniciais obtidas na ETAPA-II não foram modificadas durante a ETAPA-III, uma vez que todos os nós desconhecidos utilizaram mais do que dois nós de referência para estimar sua posição inicial, não atendendo à condição *(i)* do Algoritmo 5, definida na Seção 4.2.3. De acordo com a Figura 44, em todas as configurações de enxame consideradas nesta seção, a redução da $MMEP$ proporcionada pelo BSA ocorre predominantemente até $j = 5$, estagnando após tal iteração.

A Figura 45 apresenta as $MMEPs$ alcançadas ao fim da ETAPA-II, pelo Min-Max, e ao fim da ETAPA-IV pelos dois algoritmos em questão. Os IC das $MMEPs$ apresentadas na Figura 45 são de 95%. Considerando os experimentos com 10% de nós de referência, o BSA-HTA foi capaz de reduzir a $MMEP$ inicial, obtida pelo Min-Max, entre 44,9% e 58,1%. Já com o BSA-MMA, a redução foi entre de 36,2% e 56,4%. Para esta proporção de nós de referência, a menor $MMEP$ final obtida pelos algoritmos BSA-HTA e BSA-MMA foi de $12,8mm$ e $13,3mm$, respectivamente, ambas em enxames de 40 robôs. Note que, para enxames com 10% de nós de referência, ambos os algoritmos



(a) BSA-HTA (b) BSA-MMA

Figura 44: Evolução da MMEP em enxames de diferentes tamanhos

tiveram seu desempenho degradado com o aumento do tamanho do enxame. Sob este aspecto, o BSA-MMA se mostrou mais sensível ao aumento do tamanho do enxame do que o BSA-HTA.

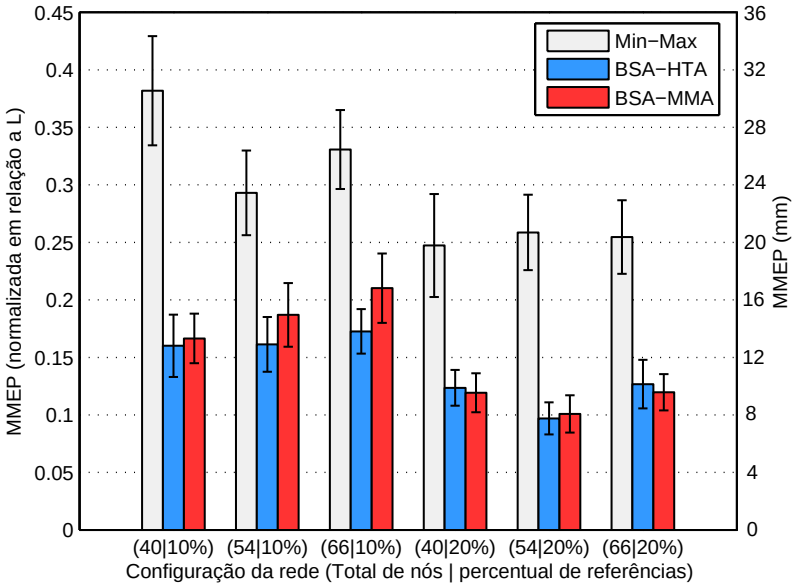


Figura 45: Resultados obtidos quanto a diferentes tamanhos de enxame

Considerando os casos com 20% de nós de referência, o BSA-HTA reduziu a $MMEP$ inicial, obtida pelo Min-Max, entre 50,0% e 62,5%. Já o BSA-MMA reduziu a $MMEP$ alcançada pelo Min-Max entre 51,8% e 61,0%. Para esta proporção de nós de referência, a menor $MMEP$ final, obtida pelos algoritmos BSA-HTA e BSA-MMA, foi

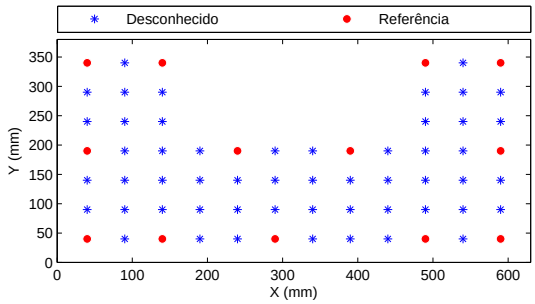
de $7,8mm$ e $8,1mm$, respectivamente, ambas em enxames de 54 robôs. Nestes experimentos, com 20% de nós de referência, o algoritmo BSA-MMA apresentou desempenho similar ao BSA-HTA. Cabe ressaltar que, pelos resultados obtidos, o aumento do tamanho do enxame não impactou significativamente no desempenho dos algoritmos.

É digno de nota que o aumento da proporção de nós de referência na rede tende a melhorar o desempenho dos algoritmos. Considerando os três tamanhos de enxame, o aumento da proporção de referências de 10% para 20% reduziu a $MMEP$ final do BSA-HTA de 22,8% a 39,9%. No caso do BSA-MMA, a redução da $MMEP$ final foi de 28,3% a 46,1%.

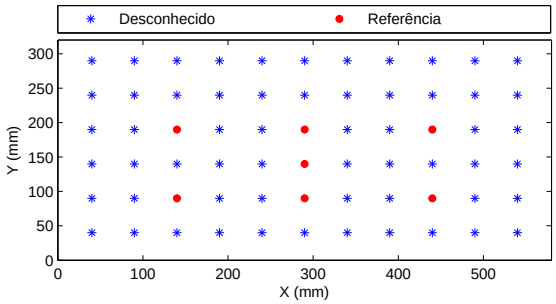
5.2.3 Avaliação Quanto a Diferentes Topologias

Os algoritmos BSA-HTA e BSA-MMA foram executados em enxames organizados de acordo com as quatro topologias apresentadas na Figura 46 e retratadas na Figura 47. A Figura 46a apresenta um enxame com topologia em malha na forma de um C. A Figura 46b apresenta uma topologia em malha retangular onde todos os nós de referência estão no interior do enxame, *i.e.* os nós desconhecidos não estão todos cercados por nós de referência. A Figura 46c apresenta um enxame com topologia em malha, contendo um obstáculo no centro. A Figura 46d apresenta um enxame com ausência de topologia, onde os robôs foram posicionados com base nas regras especificadas por (LANGENDOEN; REIJERS, 2003) e apresentadas na Seção 5.1.1.

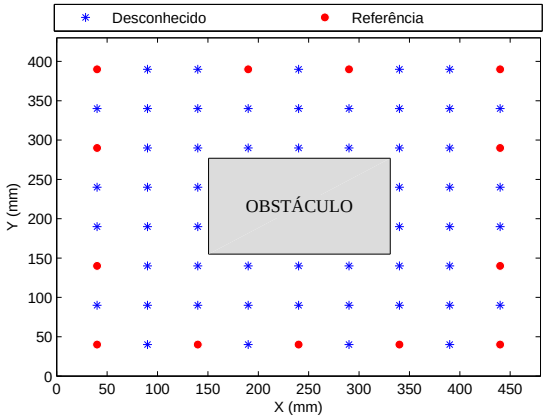
Nas topologias das Figuras 46a, 46c, 46d, fixamos o tamanho do enxame em 66 robôs e a proporção de nós de referência em 19,7% (13 nós de referência), permitindo a comparação dos resultados com aqueles obtidos pela configuração em malha retangular (66|20%) da Seção 5.2.2. Cabe ressaltar que na Figura 46d, os nós de referência foram escolhidos com base nas regras especificadas por (LANGENDOEN; REIJERS, 2003) e apresentadas na Seção 5.1.1. Neste caso, 9 referências foram escolhidas com base em uma malha de 3×3 vértices, e as 4 referências remanescentes foram escolhidas aleatoriamente, totalizando os 13 nós de referência. Na topologia da Figura 46b, adotamos um enxame de 66 robôs com 10,6% de nós de referência (ou seja, 7 nós de referência), permitindo a comparação dos resultados com aqueles obtidos pela configuração em malha retangular (66|10%) da Seção 5.2.2.



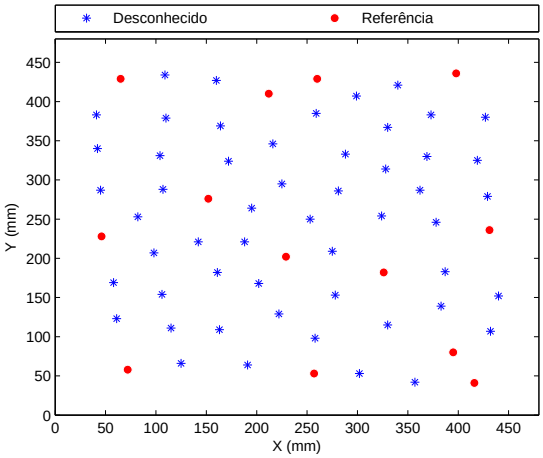
(a) Malha na forma de um C



(b) Malha retangular com referências internas



(c) Malha com obstáculo

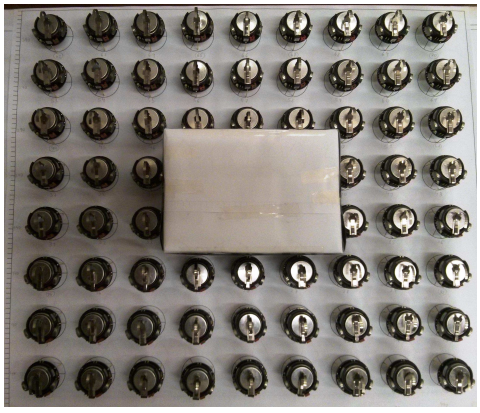


(d) Ausência de topologia

Figura 46: Diferentes topologias avaliadas nos experimentos

(a) Malha na forma de um C

(b) Malha retangular com referências internas



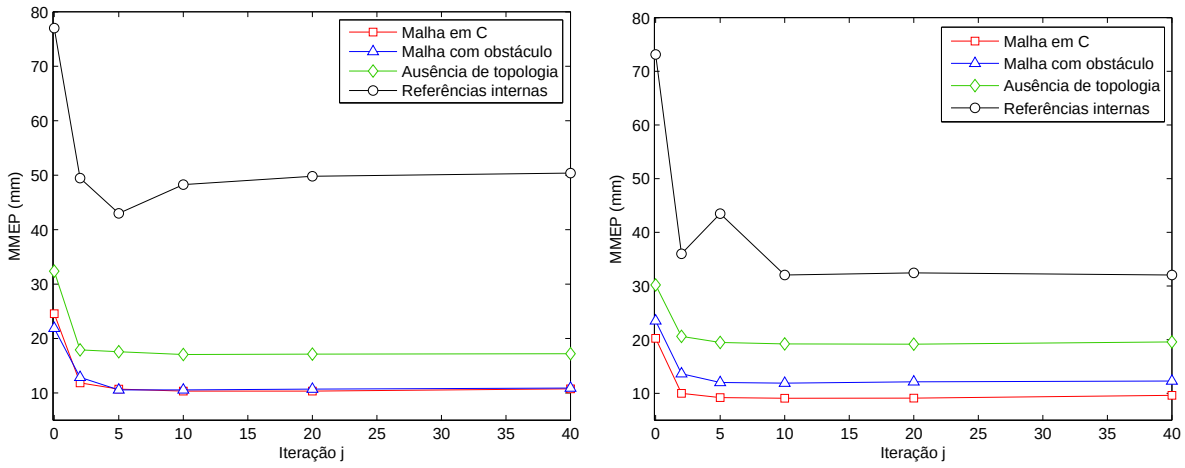
(c) Malha com obstáculo

(d) Ausência de topologia

Figura 47: Enxames de *Kilobots* com diferentes topologias

Cabe ressaltar que, em todos os experimentos desta seção, a distância limite de medição foi configurada como $L = 80mm$. Para cada topologia apresentada na Figura 46, foram realizados quatro experimentos, sendo dois com o BSA-HTA e dois com o BSA-MMA.

A evolução do processo de localização pode ser observada na Figura 48, onde são apresentadas as $MMEPs$ (mm) obtidas ao longo das iterações j da ETAPA-IV do Algoritmo 7. As $MMEPs$ apresentadas na iteração 0 correspondem aos resultados obtidos pelo Min-Max ao término da ETAPA-II. Cabe lembrar que, nestes experimentos, as posições iniciais obtidas na ETAPA-II não foram modificadas durante a ETAPA-III, uma vez que todos os nós desconhecidos utilizaram mais do que dois nós de referência para estimar sua posição inicial, não atendendo à condição *(i)* do Algoritmo 5. De acordo com a Figura 48, nestas topologias, a otimização da $MMEP$ ocorre predominantemente até $j = 5$, estagnando após esta iteração. Entretanto, a topologia com referências internas constitui uma exceção à tal assertiva, uma vez que, neste caso, o processo de otimização só estagna após $j = 10$.



(a) BSA-HTA (b) BSA-MMA

Figura 48: Evolução da MMEP em diferentes topologias

Na Figura 49, os resultados obtidos nas topologias de malha em C, de malha com obstáculo e com ausência de topologia são comparados com aqueles obtidos com a configuração de malha retangular (66|20%), apresentados na Seção 5.2.2. Nesta figura são apresentadas as $MMEPs$ alcançadas ao fim da ETAPA-II, pelo Min-Max, e ao fim da ETAPA-IV dos algoritmos BSA-HTA e BSA-MMA. Os IC das $MMEPs$ apresentadas na Figura 49 são de 95%.

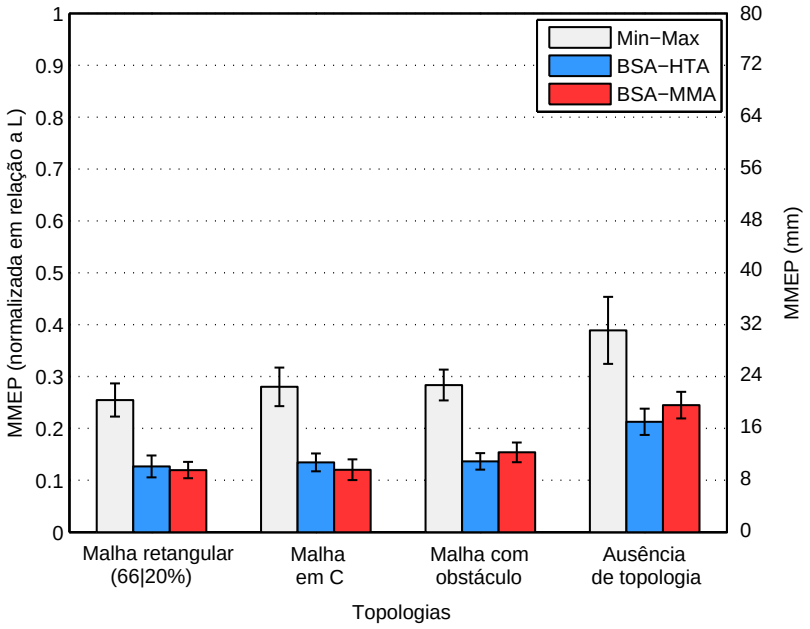


Figura 49: Resultados obtidos com diferentes topologias

Considerando as topologias abordadas na Figura 49, o BSA-HTA foi capaz de reduzir a $MMEP$ inicial, obtida pelo Min-Max, entre 45,0% e 52,0%. Já no caso do BSA-MMA, a redução da $MMEP$ foi entre de 37,4% e 57,0%. Observamos que os melhores resultados foram obtidos na topologia de malha retangular, tendo o BSA-HTA e o BSA-MMA alcançado as $MMEPs$ de $10,8mm$ e $9,6mm$, respectivamente. Nas topologias de malha em C e de malha com obstáculo, os algoritmos BSA-HTA e BSA-MMA apresentaram desempenhos ligeiramente inferiores aos obtidos na topologia de malha retangular. Entretanto, nestas topologias, as $MMEPs$ finais do BSA-HTA e do BSA-MMA não excederam $1mm$ e $3mm$ do valor obtido na topologia em malha retangular, respectivamente. No caso do enxame com ausência de topologia, a degradação das $MMEP$ foi mais expressiva. Na ausência de topologia, comparando com a topologia de malha retangular, as $MMEPs$ do BSA-HTA e BSA-MMA tiveram um aumento de $7mm$ e $10mm$, respectivamente. Da mesma forma que nas simulações apresentadas na Seção 5.1.3, verificamos nestes experimentos uma relação entre a conectividade do enxame e o desempenho dos algoritmos. A Tabela 8 apresenta a conectividade média de cada uma das referidas topologias, bem como os valores de $MMEP$ obtidos por cada algoritmo. É possível observar, a partir destes dados, quer as $MMEPs$ tendem a aumentar com a redução da conectividade do enxame.

Tabela 8: Conectividade média e $MMEPs$ obtidas em diferentes topologias

Topologia	$MMEP$ (mm)		Conectividade média
	BSA-HTA	BSA-MMA	
Malha retangular	10,14	9,57	6,98
Malha em C	10,76	9,63	6,62
Malha com obstáculo	10,91	12,31	6,55
Ausência de topologia	17,20	19,59	5,68

A Figura 50 apresenta uma comparação entre os resultados obtidos na topologia de malha com referências internas, com aqueles obtidos na malha retangular (66|10%), apresentada na Seção 5.2.2, na qual todos os nós desconhecidos estão cercados por nós de referência. Os IC das $MMEPs$ apresentadas na Figura 50 são de 95%. Ressaltamos que ambas as topologias possuem o mesmo número de nós desconhecidos e de referência, a mesma distribuição de nós e, portanto, a mesma conectividade. As duas topologias diferem apenas na disposição dos nós de referência.

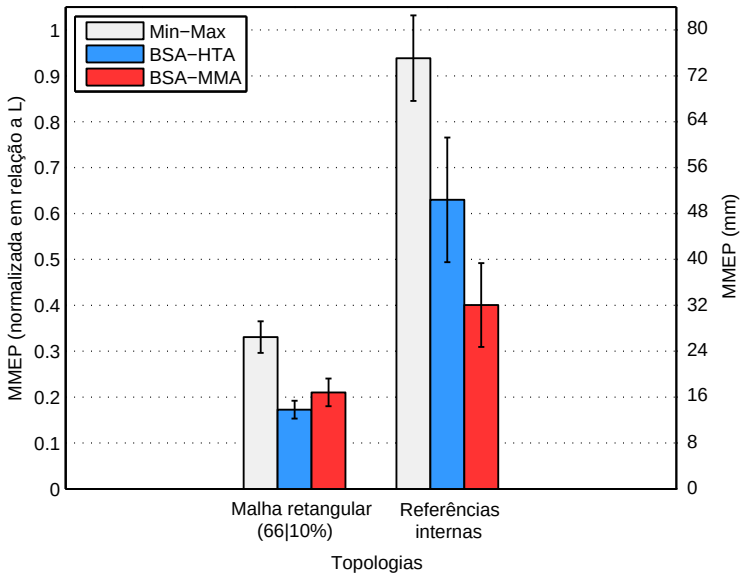
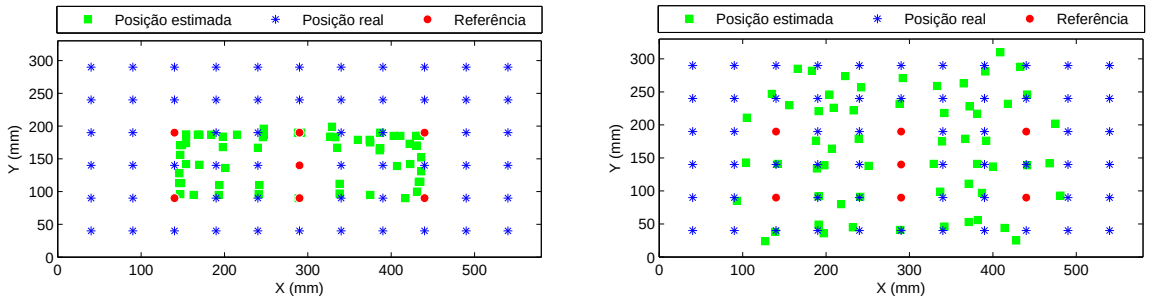


Figura 50: Comparação de desempenho referente ao posicionamento das referências

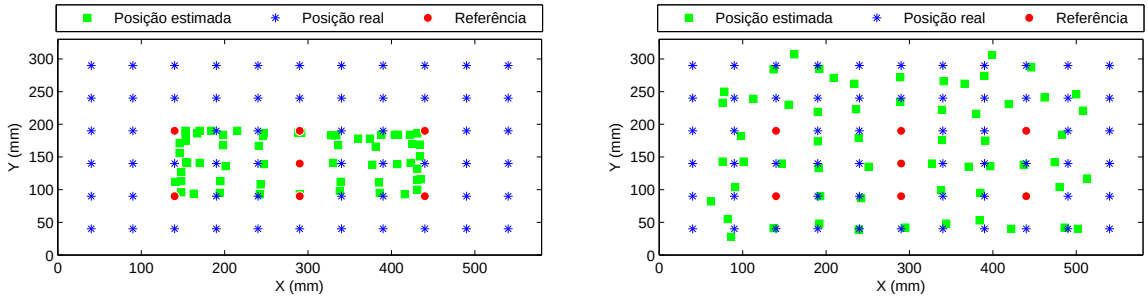
Pelo exposto na Figura 50, verificamos que, apesar da similaridade dos enxames, a disposição dos nós de referência influencia no desempenho dos algoritmos BSA-HTA e BSA-MMA. Note que, na topologia com referências internas, a $MMEP$ do Min-Max foi 184% maior do que a obtida na malha retangular cercada por nós de referência. Cabe lembrar que, conforme apresentado na Seção 4.2.2 e verificado nas simulações da Seção 5.1.3, este aumento decorre do fato de que todas as posições estimadas pelo Min-Max residem sempre dentro de uma área delimitada pelos nós de referência. Este fato tam-

bém pode ser verificado nas Figuras 51a e 52a, que apresentam exemplos de resultados obtidos pelo Min-Max ao término da ETAPA-II dos algoritmos BSA-HTA e BSA-MMA, respectivamente. Nas Figuras 51b e 52b, podemos constatar que os algoritmos BSA-HTA e BSA-MMA foram capazes de transpor a limitação imposta na ETAPA-II pelo Min-Max, reduzindo o erro de posição dos nós que estão fora da área delimitada pelos nós de referência. Verificamos, contudo, que ao fim da ETAPA-IV, em ambos os algoritmos, os nós mais afastados da área onde estão as referências tendem a ter um maior erro de posição. Os resultados apresentados na Figura 50 reiteram, tal como reportado nas simulações da Seção 5.1.3, o melhor desempenho do BSA-MMA em redes com referências internas, quando comparado ao BSA-HTA. Nestes experimentos, a $MMEP$ final do BSA-MMA foi 36,4% menor do que a $MMEP$ obtida pelo BSA-HTA. Tal diferença de desempenho também pode ser observada nas Figuras 51b e 52b, especialmente no que se refere aos nós que não estão cercados por referências.



(a) ETAPA-II (Min-Max) (b) ETAPA-IV (BSA-HTA)

Figura 51: Evolução do processo de localização com o algoritmo BSA-HTA na topologia de malha com referências internas

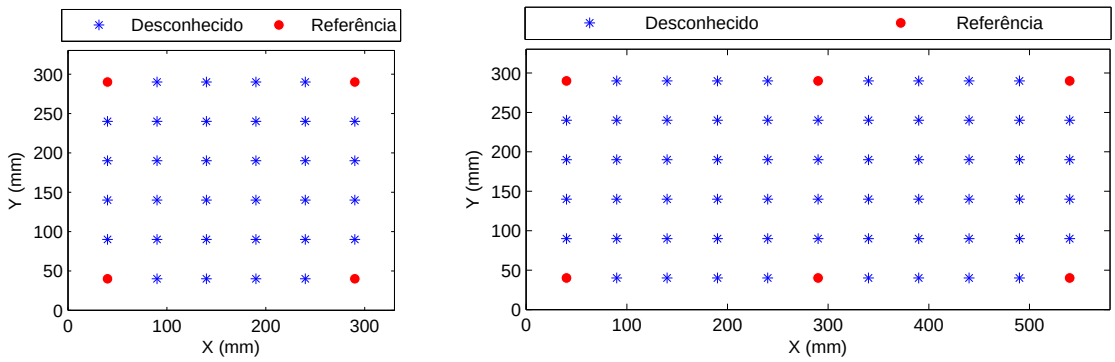


(a) ETAPA-II (Min-Max) (b) ETAPA-IV (BSA-MMA)

Figura 52: Evolução do processo de localização com o algoritmo BSA-MMA na topologia de malha com referências internas

5.2.4 Avaliação do Método MCMM

Para avaliar o desempenho do método MCMM, foram realizados experimentos em duas configurações de enxame contendo aproximadamente 10% de nós de referência. A primeira configuração consiste em uma topologia em malha quadrada composta por 32 nós desconhecidos e 4 nós de referência. Nesta configuração, os nós de referência são posicionados nas extremidades do enxame, conforme apresentado na Figura 53a. A segunda configuração, apresentada na Figura 53b, possui 60 nós desconhecidos e 6 nós de referência, e equivale a dois enxames da Figura 53a posicionados lado a lado. Em todos os experimentos desta seção, a distância limite de medição foi configurada para $L = 70mm$. A fim de simular restrições quanto à divulgação dos nós de referência, foi imposto o máximo de 5 saltos para retransmissão de mensagens da ETAPA-I, fazendo com que os nós desconhecidos não fossem capazes de reconhecer referências além deste limite.



(a) Enxame com 32 nós desconhecidos e 4 nós de referência

(b) Enxame com 60 nós desconhecidos e 6 nós de referência

Figura 53: Configurações de enxame utilizadas para avaliação do MCMM

Assim como na Seção 5.1.4, apenas o algoritmo BSA-MMA foi utilizado nos experimentos desta seção, tendo em vista que os resultados obtidos pelo Min-Max e pelo MCMM independem do algoritmo de otimização e do fator de confiança empregados na ETAPA-IV. Para cada configuração de enxame apresentada na Figura 53, foram realizados dois experimentos e calculada a $MMEP$ ao fim das ETAPAS-II, III e IV. As $MMEPs$ obtidas são apresentadas na Figura 54, junto com os seus IC de 95%. É apresentada ainda, na Figura 54, a $MMEP$ obtida ao fim da ETAPA-IV do BSA-MMA, quando aplicado sem a execução do MCMM na ETAPA-III.

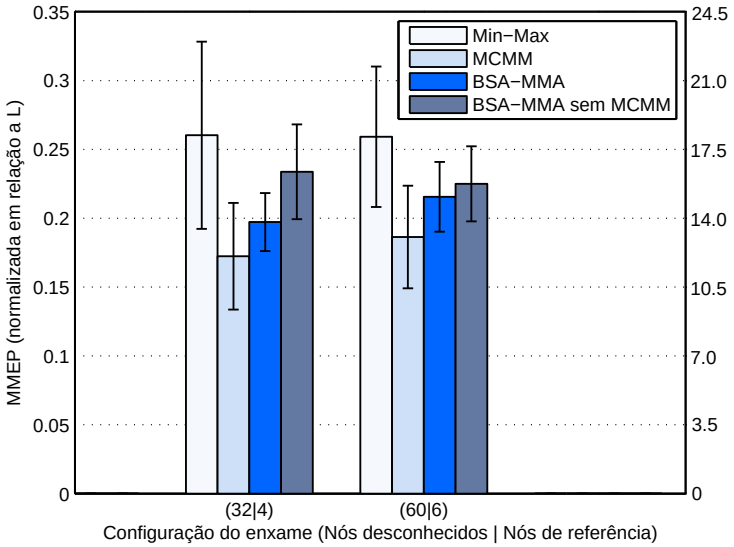
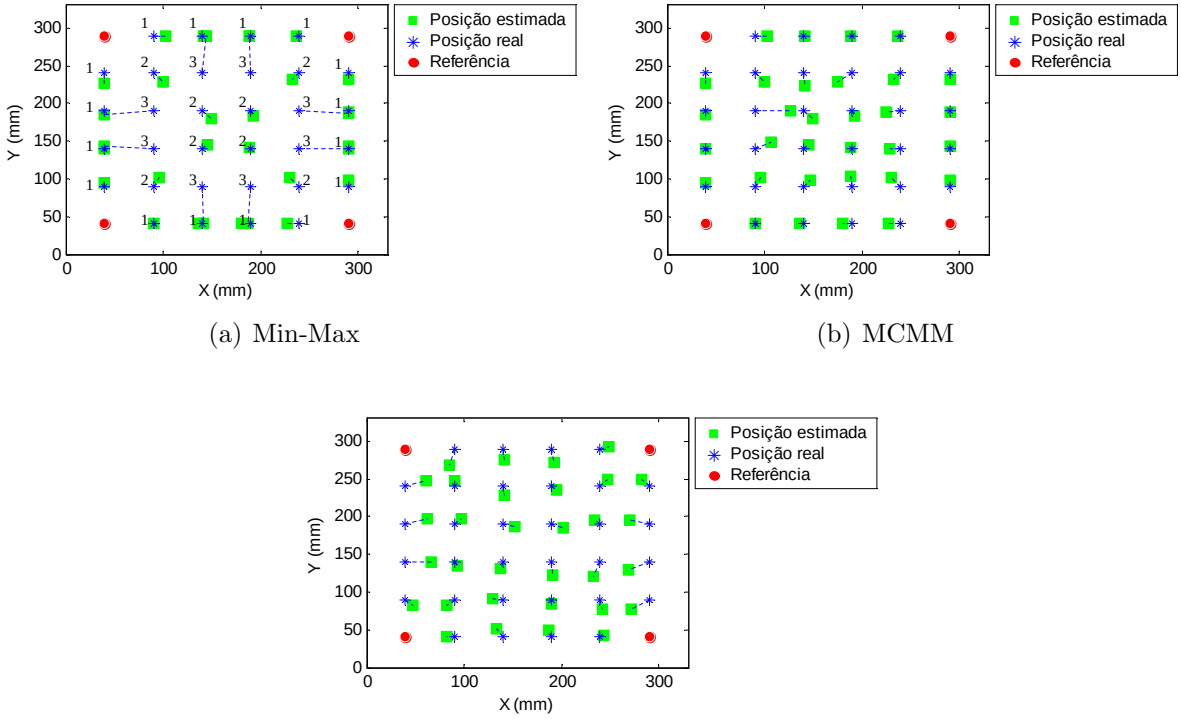


Figura 54: Desempenho do algoritmo em redes com restrições quanto à divulgação dos nós de referência

Podemos observar que, em ambas as redes, o desempenho do Min-Max foi similar, com $MMEPs$ de aproximadamente $18mm$. Após a execução do MCMM, a rede com 32 nós desconhecidos e 4 nós de referência teve sua $MMEP$ reduzida para $11,5mm$, o que representa uma melhoria de 36%. Já no caso da rede com 60 nós desconhecidos e 6 nós de referência, a $MMEP$ foi reduzida para $13mm$, *i.e.* 28% a menos que no Min-Max, o que indica um desempenho ligeiramente inferior ao obtido pela rede com 32 nós desconhecidos e 4 nós de referência. Constatamos ainda que, nestas configurações de enxame, a $MMEP$ obtida ao fim da ETAPA-IV pelo BSA, apesar de inferior àquela obtida pelo Min-Max, foi superior à obtida pelo $MCMM$.

A evolução do processo de localização destes enxames pode ser observada nos exemplos das Figuras 55 e 56. Nas Figuras 55a e 56a, os nós identificados pelo número 1 conheceram apenas duas referências, mas, de acordo com o Algoritmo 5, não tiveram suas posições atualizadas pelo MCMM pelo fato de terem um alinhamento inferior ao limiar estabelecido de $\beta = -0,05$. Os nós identificados pelo número 2 conheceram mais do que 2 nós de referência durante a ETAPA-I e, portanto, não tiveram suas posições atualizadas pelo MCMM, conforme determina o Algoritmo 5. Note que o comportamento apresentado pelos nós 1 e 2 é desejável, tendo em vista a boa qualidade da posição inicial estimada por estes nós. Os nós identificados pelo número 3 conheceram apenas duas referências durante a ETAPA-I e apresentaram alinhamento superior ao limiar de $\beta =$

$-0,05$. Neste caso, conforme estabelece o Algoritmo 5, os nós tiveram suas posições atualizadas pelo MCMM. Tal comportamento também é desejável, tendo em vista que a posição inicial estimada pelos nós número 3 foi degradada pela restrição imposta na divulgação dos nós de referência. Um comportamento adverso foi apresentado pelo nó identificado pelo número 4 na Figura 56a. Este nó, cuja posição inicial apresentou boa precisão, teve a posição modificada pelo MCMM. Apesar de este nó estar fisicamente bem alinhado com duas referências, seu alinhamento calculado foi superior ao limiar de $\beta = -0,05$, em virtude de um maior erro nas suas medidas de distância. Cabe ressaltar que, apesar do comportamento adverso do nó número 4 do exemplo, o desempenho global do MCMM foi positivo, conforme as $MMEPs$ apresentadas na Figura 54. Após a execução da ETAPA-IV, as posições estimadas pelos nós número 3 e pelo nó número 4, de uma forma geral, se aproximaram das suas posições reais. Entretanto, após a ETAPA-IV, as posições estimadas pelos nós 1 e 2 se afastaram das posições reais causando, nestas topologias, o aumento da $MMEP$ observado na Figura 54, quando comparado aos resultados obtidos na ETAPA-III pelo MCMM.

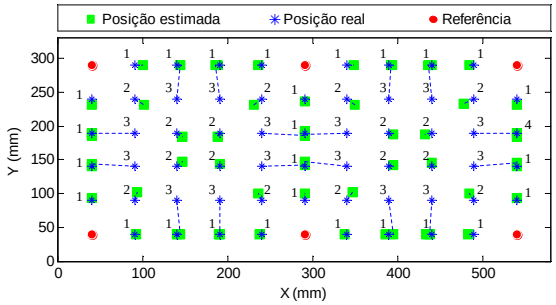


(a) Min-Max

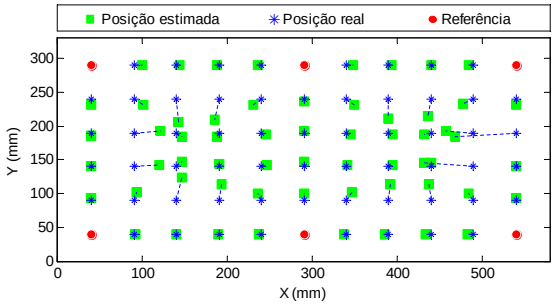
(b) MCMM

(c) BSA-MMA

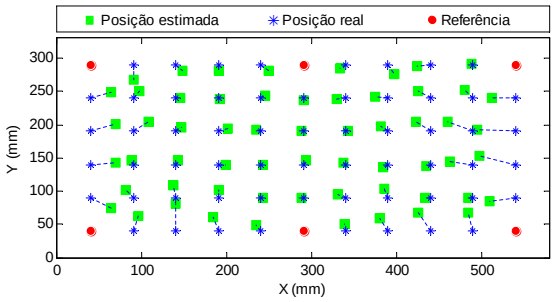
Figura 55: Localização em redes com 32 nós desconhecidos e 4 referências



(a) Min-Max



(b) MCMM



(c) BSA-MMA

Figura 56: Localização em redes com 60 nós desconhecidos e 6 referências

No que se refere ao resultado final do processo de localização, constatamos, pela Figura 54, que a $MMEP$ obtida pelo BSA-MMA, com a execução do MCMM, foi melhor do que a $MMEP$ obtida sem a execução do MCMM, sendo a diferença mais expressiva nos experimentos com 32 nós desconhecidos e 4 referências.

5.3 Considerações Finais

Neste capítulo foram avaliados quatro algoritmos de localização: PSO-HTA, PSO-MMA, BSA-HTA e BSA-MMA. Os resultados obtidos indicam que, de uma forma geral, os quatro algoritmos propostos possuem melhor desempenho que os algoritmos *N-Hop Multilateration* (SAVVIDES; PARK; SRIVASTAVA, 2002) e *Robust Positioning* (RABAEY; LANGENDOEN, 2002), quando submetidos às simulações especificadas por (LANGENDOEN; REIJERS, 2003). Nestas simulações, o BSA-HTA foi o algoritmo que obteve o melhor desempenho, com $MMEPs$ 40% a 51% menores do que os algoritmos *N-Hop Multilateration* e *Robust Positioning*.

Os resultados indicam que o aumento da conectividade da rede melhora o desempenho dos algoritmos, ao custo de um maior tempo de processamento. Da mesma forma,

o desempenho dos algoritmos propostos melhora com o aumento da proporção de nós de referência na rede.

No que tange aos algoritmos de otimização avaliados nas simulações, os algoritmos baseados no BSA apresentaram, na maioria dos casos, resultados melhores que os algoritmos baseados no PSO, em tempos de processamento similares. A diferença entre as $MMEPs$ obtidas pelo BSA e pelo PSO foi mais expressiva nos casos de topologias com referências internas. Nos demais casos, a vantagem do BSA foi menos significativa. No que se refere às diferentes topologias avaliadas, observou-se melhor desempenho nas formas de enxame que possuem menor número de nós desconhecidos no perímetro. De uma forma geral, nas simulações com diferentes topologias, o BSA-HTA foi o algoritmo que obteve o melhor desempenho. Nos experimentos, os algoritmos BSA-HTA e BSA-MMA obtiveram desempenhos similares.

No que diz respeito à posição relativa dos nós de referência, os algoritmos obtiveram melhor desempenho em enxames cercados por tais nós. Cabe ressaltar, entretanto, a capacidade que os algoritmos apresentaram em localizar nós desconhecidos que não estejam cercados por nós de referência, especialmente o BSA-MMA, tanto nas simulações quanto nos experimentos. Todavia, essa capacidade diminui à medida que os nós desconhecidos se afastam da região onde estão concentrados os nós de referência.

Referente aos fatores de confiança, de uma forma geral, o HTA-CF obteve um melhor desempenho que o MMA-CF. Entretanto, o MMA-CF é mais adequado para uso em redes com baixo erro de medição de distância e poucos nós desconhecidos, bem como em redes com referências internas.

Com relação ao MCMM, o método se mostrou eficaz na redução da $MMEP$ em situações onde há restrição na divulgação dos nós de referência. Nas simulações, o método proporcionou 30% a 65% de redução da $MMEP$ obtida pelo Min-Max, sendo mais eficiente em redes com maior densidade de nós desconhecidos. Nos experimentos, a redução da $MMEP$ foi de 28% a 36%. Constatamos ainda que, nestes casos, o resultado final do processo de localização é, em geral, melhor quando o MCMM é executado na ETAPA-III do que quando o mesmo não é executado.

Capítulo 6

CONCLUSÕES E TRABALHOS FUTUROS

NESTE capítulo são apresentadas as principais conclusões acerca do método de localização colaborativa de SRSs, proposto nesta dissertação, bem como conclusões relativas ao desempenho dos quatro algoritmos resultantes do referido método. Adicionalmente, são apresentadas algumas possíveis melhorias, assim como algumas possibilidades de trabalhos futuros.

6.1 Conclusões

Esta dissertação abordou o problema de localização de robôs em SRSs. O trabalho desenvolvido propõe um método de localização colaborativa como uma alternativa ao uso de receptores GPS, em situações em que o uso de tal sistema é inviável.

O método é dividido em quatro etapas. Nas ETAPAS-I e II, são utilizados os algoritmos Sum-Dist e Min-Max, para medir distâncias em múltiplos saltos e para estimar a posição inicial dos nós desconhecidos, respectivamente. Na ETAPA-III é utilizado o método MCMM (SÁ; NEDJAH; MOURELLE, 2015c), ora proposto, com o objetivo de melhorar a qualidade da posição inicial dos nós desconhecidos em caso de falhas durante o reconhecimento dos nós de referência na ETAPA-I. Na ETAPA-IV, a posição dos nós desconhecidos é refinada com base em duas alternativas de algoritmo de otimização por inteligência coletiva: o PSO e o BSA. Para compor a função objetivo, é introduzido um novo método para o cálculo do fator de confiança dos nós da rede, o MMA-CF (SÁ; NEDJAH; MOURELLE, 2014b), o qual é comparado com o fator de confiança HTA-CF, proposto por (EKBERG, 2009).

Com base no método de localização proposto, foram desenvolvidos quatro algoritmos (PSO-HTA, PSO-MMA, BSA-HTA e BSA-MMA), os quais foram avaliados por meio de simulações realizadas no MATLAB® e experimentos conduzidos em exames de robôs do tipo *Kilobot*.

Primeiramente, foram realizadas simulações visando à comparação do desempenho dos quatro algoritmos com o desempenho obtido pelos algoritmos *N-Hop Multilateration* e *Robust Positioning* em (LANGENDOEN; REIJERS, 2003). De uma forma geral, os quatro algoritmos apresentaram melhor desempenho que os algoritmos *N-Hop Multilateration* e *Robust Positioning*, indicando a eficácia do método proposto. Nestas simulações, o algoritmo que obteve o melhor desempenho foi o BSA-HTA, com desempenho 40% a 51% melhor do que os algoritmos *N-Hop Multilateration* e *Robust Positioning*.

Posteriormente, foram realizadas simulações e experimentos, explorando problemas com diferentes topologias, quantidades de nós e proporção de nós de referência. Nas simulações, os algoritmos baseados no BSA apresentaram resultados melhores que o PSO, com tempos de processamento aproximadamente iguais. Cabe ressaltar que a vantagem do BSA foi mais expressiva nos casos de topologias com referências internas, sendo menos significativa nos demais casos.

No que diz respeito às diferentes topologias, os algoritmos obtiveram melhor desempenho nas formações com menor proporção de nós desconhecidos no perímetro. Considerando todas as topologias exploradas nas simulações, o BSA-HTA foi o algoritmo que obteve o melhor desempenho. Já nos experimentos, os algoritmos BSA-HTA e BSA-MMA obtiveram desempenhos similares. Cabe ressaltar que, tanto nas simulações quanto nos experimentos, os algoritmos apresentaram boa capacidade de localizar nós desconhecidos em topologias com referências internas, *i.e.* topologias onde os nós de referência não cercam o enxame. Neste caso específico, o BSA-MMA foi o algoritmo que apresentou melhor desempenho. Esta capacidade representa um ponto positivo, uma vez que, segundo (LANGENDOEN; REIJERS, 2003), a maioria dos algoritmos de localização é muito sensível à presença, ou ausência, de referências nas extremidades da rede, e possui melhor desempenho quando as referências cercam os nós desconhecidos. Todavia, observa-se que essa capacidade diminui à medida que os nós desconhecidos se afastam da região onde estão concentrados os nós de referência. Em comparação com as topologias cercadas por referências, as com referências internas apresentaram maiores erros de posicionamento.

Os resultados obtidos indicam que o aumento da conectividade, causado pelo aumento do número de nós na rede, melhora o desempenho dos algoritmos ao custo, entretanto, de um maior tempo de processamento. Da mesma forma, o desempenho dos algoritmos melhora com o aumento da proporção de nós de referência na rede.

Com relação aos fatores de confiança, o HTA-CF obteve, na maioria dos casos, um melhor desempenho que o MMA-CF. Entretanto, o MMA-CF se mostrou mais adequado para uso em redes com referências internas, bem como em sistemas com baixo erro de medição de distância e poucos nós desconhecidos.

Referente ao MCMM, o método se mostrou capaz de contribuir com a redução da $MMEP$ em situações onde há restrição na divulgação dos nós de referência. Nas simulações, o método proporcionou 30% a 65% de redução da $MMEP$ obtida pelo Min-Max. Nos experimentos, a redução da $MMEP$ foi de 28% a 36%. Verificamos ainda que, nestas situações, o resultado final do processo de localização é, em geral, melhor quando o MCMM é executado na ETAPA-III do que quando o mesmo não é executado.

Apesar da ênfase dada à localização colaborativa em SRSs e do fato de os algoritmos terem sido implementados em robôs, entende-se que a solução proposta também é aplicável aos problemas de localização enfrentados nas WSNs.

6.2 Trabalhos Futuros

A fim de contribuir com a melhoria do método proposto, sugere-se a investigação de outros algoritmos de otimização que possam ser aplicados à ETAPA-IV, de refinamento. Outrossim, sugere-se o estudo de possíveis melhorias na modelagem da função objetivo como, por exemplo, o desenvolvimento de novas formas de cálculo do fator de confiança, inclusive que combinem os métodos MMA-CF e HTA-CF.

Na maioria dos casos, o refinamento da ETAPA-IV apresentou melhoras significativas na média do erro de posicionamento. Entretanto, houve situações em que a $MMEP$ alcançada pelo MCMM foi melhor do que a $MMEP$ obtida ao fim da ETAPA-IV dos algoritmos, como no caso dos experimentos em redes com topologia em malha da Seção 5.2.4. Logo, o desempenho dos algoritmos poderia ser melhorado caso houvesse uma forma de decidir sobre a execução ou não da etapa de refinamento. Sendo assim, encorajamos a busca por heurísticas que permitam que cada nó desconhecido seja capaz de estimar

a qualidade da posição calculada na ETAPA-III de forma que, caso esta apresente boa exatidão, não seja modificada nas etapas subsequentes.

Como trabalho futuro, sugerimos a extensão do método para problemas que envolvam mobilidade. Uma possibilidade seria repetir periodicamente a execução completa do algoritmo. Entretanto consideramos que esta solução não seria proficiente. Conforme descrito na Seção 4.2.4.2, o problema tratado na ETAPA-IV tem a característica de ser dinâmico, uma vez que a função objetivo muda ao longo do tempo, em virtude da atualização das posições dos nós vizinhos. Dessa forma, acreditamos que, até uma determinada velocidade de deslocamento, a movimentação dos nós represente para o algoritmo um comportamento similar àquele decorrente da atualização das posições que ocorre mesmo quando os nós estão estáticos. Caso esta premissa esteja correta, acreditamos que as ETAPAS-I, II e III poderiam ser executadas apenas uma vez, para estimar a posição inicial dos nós. Após isto, com os nós em movimento, poderíamos apenas manter a ETAPA-IV em execução. Entretanto, há de se investigar a necessidade de recalcular o fator de confiança dos nós durante a ETAPA-IV.

Adicionalmente incentivamos a aplicação do método em diferentes tipos de robôs ou sensores, inclusive que utilizem outras técnicas de medição de distância, bem como em robôs que ocupem espaços tridimensionais, como robôs submarinos ou voadores.

REFERÊNCIAS

AKKAYA, K.; YOUNIS, M. A survey on routing protocols for wireless sensor networks. *Ad hoc networks*, Elsevier, v. 3, n. 3, p. 325–349, 2005.

AKYILDIZ, I. F. et al. Wireless sensor networks: a survey. *Computer networks*, Elsevier, v. 38, n. 4, p. 393–422, 2002.

ATMEL. *ATmega328P*. Março 2015. Disponível em: <http://www.atmel.com>.

ATMEL. *AVR Studio*. Março 2015. Disponível em: <http://www.atmel.com>.

BENET, G. et al. Using infrared sensors for distance measurement in mobile robots. *Robotics and autonomous systems*, Elsevier, v. 40, n. 4, p. 255–266, 2002.

BONABEAU, E.; DORIGO, M.; THERAULAZ, G. *Swarm intelligence: from natural to artificial systems*. New York, NY, USA: Oxford University Press, 1999.

BULUSU, N.; HEIDEMANN, J.; ESTRIN, D. Gps-less low-cost outdoor localization for very small devices. *Personal Communications*, IEEE, v. 7, n. 5, p. 28–34, 2000.

CHAN, F. T.; TIWARI, M. K. *Swarm intelligence, focus on ant and particle swarm optimization*. Vienna, Austria: I-Tech Education and Publishing, 2007.

CHENG, B. H.; VANDENBERGHE, L.; YAO, K. Distributed algorithm for node localization in wireless ad-hoc networks. *ACM Transactions on Sensor Networks (TOSN)*, ACM, v. 6, n. 1, p. 8.1–8.20, 2009.

CIVICIOGLU, P. Backtracking search optimization algorithm for numerical optimization problems. *Applied Mathematics and Computation*, Elsevier, v. 219, n. 15, p. 8121–8144, 2013.

CIVICIOGLU, P. *BSA code for MATLAB*. Dezembro 2013. Disponível em: <http://www.pinarcivicioglu.com/bsa.html>.

DENNIS, J. E.; SCHNABEL, R. B. *Numerical methods for unconstrained optimization and nonlinear equations*. Philadelphia, PA, USA: SIAM, 1996.

DORIGO, M.; BIRATTARI, M. Ant colony optimization. In: *Encyclopedia of Machine Learning*. New York, NY, USA: Springer US, 2010. p. 36–39.

EBERHART, R.; SIMPSON, P.; DOBBINS, R. *Computational intelligence PC tools*. San Diego, CA, USA: Academic Press Professional, Inc., 1996.

EKBERG, P. *Swarm-Intelligent Localization*. Dissertação (Mestrado) — Uppsala Universitet, Uppsala, Sweden, Agosto 2009.

ENGELBRECHT, A. P. *Fundamentals of Computational Swarm Intelligence*. West Sussex, Inglaterra: John Wiley & Sons, 2006.

ENGELBRECHT, A. P. *Computational intelligence: an introduction*. 2. ed. West Sussex, Inglaterra: John Wiley & Sons, 2007.

GCTRONIC. *ELISA-III Frame*. Abril 2015. Disponível em: <http://www.gctronic.com/>.

GIROD, L.; ESTRIN, D. Robust range estimation using acoustic and multimodal sensing. In: IEEE. *Intelligent Robots and Systems, Proceedings of 2001 IEEE/RSJ International Conference on*. Maui, HI, USA, 2001. v. 3, p. 1312–1320.

KAKALIS, N. M.; VENTIKOS, Y. Robotic swarm concept for efficient oil spill confrontation. *Journal of hazardous materials*, Elsevier, v. 154, n. 1, p. 880–887, 2008.

KENNEDY J. E EBERHART, R. Particle swarm optimization. In: IEEE. *Proceedings of 1995 IEEE International Conference on Neural Networks*. Perth, WA, USA, 1995. v. 4, p. 1942–1948.

KERNIGHAN, B. W.; RITCHIE, D. M.; EJEKLINT, P. *The C programming language*. USA: Prentice-Hall, 1988.

KIM, S.; KWON, O. Location estimation based on edge weights in wireless sensor networks. *Journal of Korea Information and Communication Society*, v. 30, n. 10A, p. 155–162, 2005.

KORBA, L.; ELGAZZAR, S.; WELCH, T. Active infrared sensors for mobile robots. *IEEE Transactions on Instrumentation and Measurement*, IEEE, v. 43, n. 2, p. 283–287, 1994.

LANGENDOEN, K.; REIJERS, N. Distributed localization in wireless sensor networks: a quantitative comparison. *Computer Networks*, Elsevier, v. 43, n. 4, p. 499–518, 2003.

LEHNERT, C.; CORKE, P. μAV-Design and implementation of an open source micro quadrotor. In: ARAA. *Proceedings of Australasian Conference on Robotics and Automation*. Sydney, Australia, 2013. p. 1–8.

LI, L.; HALPERN, J. Y. Minimum-energy mobile wireless networks revisited. In: IEEE. *ICC 2001. IEEE International Conference on Communications*. Helsinki, Finlândia, 2001. v. 1, p. 278–283.

LIBERTI, L. et al. Euclidean distance geometry and applications. *SIAM Review*, SIAM, v. 56, n. 1, p. 3–69, 2014.

LYMBEROPOULOS, D.; LINDSEY, Q.; SAVVIDES, A. An empirical characterization of radio signal strength variability in 3-d ieee 802.15. 4 networks using monopole antennas. In: *Wireless Sensor Networks*. Zurich, Suíça: Springer, 2006. p. 326–341.

MAO, G.; FIDAN, B.; ANDERSON, B. Wireless sensor network localization techniques. *Computer networks*, Elsevier, v. 51, n. 10, p. 2529–2553, 2007.

MATHWORKS, I. *MATLAB: the language of technical computing. Desktop tools and development environment, version 7*. Michigan, USA: MathWorks, 2005.

MONDADA, F. et al. The e-puck, a robot designed for education in engineering. In: IPCB: INSTITUTO POLITÉCNICO DE CASTELO BRANCO. *Proceedings of the 9th conference on autonomous robot systems and competitions*. Castelo Branco, Portugal, 2009. v. 1, n. 1, p. 59–65.

NAWAZ, S. et al. An underwater robotic network for monitoring nuclear waste storage pools. In: *Sensor Systems and Software*. Pisa, Itália: Springer, 2010. p. 236–255.

NICULESCU, D.; NATH, B. Ad hoc positioning system (aps). In: IEEE. *Global Telecommunications Conference, 2001. GLOBECOM'01. IEEE*. San Antonio, TX, USA, 2001. v. 5, p. 2926–2931.

NICULESCU, D.; NATH, B. Localized positioning in ad hoc networks. *Ad Hoc Networks*, Elsevier, v. 1, n. 2, p. 247–259, 2003.

NOVOTNY, P. M.; FERRIER, N. J. Using infrared sensors and the phong illumination model to measure distances. In: IEEE. *Proceedings of 1999 IEEE International Conference on Robotics and Automation*. Detroit, MI, USA, 1999. v. 2, p. 1644–1649.

POLI, R. Analysis of the publications on the applications of particle swarm optimisation. *Journal of Artificial Evolution and Applications*, Hindawi Publishing Corp., v. 2008, p. 3:1–3:10, 2008.

PRIYANTHA, N. B.; CHAKRABORTY, A.; BALAKRISHNAN, H. The cricket location-support system. In: ACM. *Proceedings of the 6th annual international conference on Mobile computing and networking*. New York, NY, USA, 2000. p. 32–43.

RABAEY, C. S. J.; LANGENDOEN, K. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In: USENIX. *USENIX technical annual conference*. Monterey, CA, USA, 2002. p. 317–327.

RODIC, A.; KATIE, D.; MESTER, G. Ambient intelligent robot-sensor networks for environmental surveillance and remote sensing. In: IEEE. *Intelligent Systems and Informatics, 2009. SISY'09. 7th International Symposium on*. Subotica, Sérvia, 2009. p. 39–44.

RODOPLU, V.; MENG, T. H. Minimum energy mobile wireless networks. *Selected Areas in Communications, IEEE Journal on*, IEEE, v. 17, n. 8, p. 1333–1344, 1999.

ROMER, K.; MATTERN, F. The design space of wireless sensor networks. *Wireless Communications*, IEEE, v. 11, n. 6, p. 54–61, 2004.

RUBENSTEIN, M. *Self-assembly and self-healing for robotic collectives*. Tese (Doutorado) — University of Southern California, California, USA, 2009.

RUBENSTEIN, M.; AHLER, C.; NAGPAL, R. Kilobot: A low cost scalable robot system for collective behaviors. In: IEEE. *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. Saint Paul, MN, USA, 2012. p. 3293–3298.

SÁ, A. O. de; NEDJAH, N.; MOURELLE, L. de M. Distributed efficient localization in swarm robotics using min-max and particle swarmoptimization. In: LA-CCI. *Proceedings of The Latin American Congress on Computational Intelligence*. San Carlos de Bariloche, Argentina, 2014. v. 1, p. 7–12.

SÁ, A. O. de; NEDJAH, N.; MOURELLE, L. de M. Distributed efficient node localization in wireless sensor networks using the backtracking search algorithm. In: *Algorithms and Architectures for Parallel Processing*. Dalian, China: Springer, 2014, (Lecture Notes in Computer Science, v. 8630). p. 794–808.

SÁ, A. O. de; NEDJAH, N.; MOURELLE, L. de M. Distributed efficient localization in swarm robotic systems using swarm intelligence algorithms. *Neurocomputing*, Elsevier, p. aceito para publicação, 2015.

SÁ, A. O. de; NEDJAH, N.; MOURELLE, L. de M. Genetic and backtracking search optimization algorithms applied to localization problems. *International Journal of Innovative Computing and Applications*, Inderscience, p. aceito para publicação, 2015.

SÁ, A. O. de; NEDJAH, N.; MOURELLE, L. de M. Multi-hop collaborative min-max localization. In: IEEE. *Proceedings of The VI Latin American Symposium on Circuits and Systems - LASCAS 2015*. Montevidéu, Uruguai, 2015.

SÁ, A. O. de et al. Localização de robôs utilizando inteligência de enxame. In: SBA. *XX Congresso Brasileiro de Automática (CBA)*. Belo Horizonte, MG, Brasil, 2014. p. 9–15.

SÁ, A. O. de; NEDJAH, N.; MOURELLE, L. M. Genetic and backtracking search optimization algorithms applied to localization problems. In: *Computational Science and Its Applications - ICCSA 2014*. Guimarães, Portugal: Springer International Publishing, 2014, (Lecture Notes in Computer Science, v. 8583). p. 738–746.

ŞAHIN, E. Swarm robotics: From sources of inspiration to domains of application. In: *Swarm robotics*. Santa Monica, CA, USA: Springer, 2005, (Lecture Notes in Computer Science, v. 3342). p. 10–20.

SAVVIDES, A.; PARK, H.; SRIVASTAVA, M. B. The bits and flops of the n-hop multilateration primitive for node localization problems. In: ACM. *Proceedings of the*

1st ACM international workshop on Wireless sensor networks and applications. Atlanta, GA, USA, 2002. p. 112–121.

SHANG, Y. et al. Localization from mere connectivity. In: ACM. *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*. Annapolis, MD, USA, 2003. p. 201–212.

SHI, Y.; EBERHART, R. A modified particle swarm optimizer. In: IEEE. *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*. Anchorage, AK, USA, 1998. p. 69–73.

SICHITIU, M. L.; RAMADURAI, V.; PEDDABACHAGARI, P. Simple algorithm for outdoor localization of wireless sensor networks with inaccurate range measurements. In: ICWN. *International Conference on Wireless Networks*. Las Vegas, NV, USA, 2003. v. 2003, p. 1–6.

STELLA, M.; RUSSO, M.; BEGUŠIĆ, D. Fingerprinting based localization in heterogeneous wireless networks. *Expert Systems with Applications*, Elsevier, v. 41, n. 15, p. 6738–6747, 2014.

SUN, W.; SU, X. Wireless sensor network node localization based on genetic algorithm. In: IEEE. *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*. Xi'an, China, 2011. p. 316–319.

TESORIERO, R. et al. Improving location awareness in indoor spaces using rfid technology. *Expert Systems with Applications*, Elsevier, v. 37, n. 1, p. 894–898, 2010.

WHITEHOUSE, K.; CULLER, D. Calibration as parameter estimation in sensor networks. In: ACM. *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. Atlanta, GA, USA, 2002. p. 59–67.

WINAVR. *WinAVR*. Março 2015. Disponível em: <http://winavr.sourceforge.net/>.

XU, Y.; HEIDEMANN, J.; ESTRIN, D. Geography-informed energy conservation for ad hoc routing. In: ACM. *Proceedings of the 7th annual international conference on Mobile computing and networking*. Rome, Italy, 2001. p. 70–84.

YU, Y.; GOVINDAN, R.; ESTRIN, D. *Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks*. Los Angeles, CA, USA, 2001.

YUN, S. et al. A soft computing approach to localization in wireless sensor networks. *Expert Systems with Applications*, Elsevier, v. 36, n. 4, p. 7552–7561, 2009.

ZHANG, Q. et al. Genetic algorithm based wireless sensor network localization. In: IEEE. *Natural Computation, 2008. ICNC'08. Fourth International Conference on*. Jinan, China, 2008. v. 1, p. 608–613.

ZIOUVA, E.; ANTONAKOPOULOS, T. Csma/ca performance under high traffic conditions: throughput and delay analysis. *Computer communications*, Elsevier, v. 25, n. 3, p. 313–321, 2002.

APÊNDICE A – Resultados de Simulação

Este apêndice apresenta os valores das $MMEPs$ obtidas nas simulações apresentadas na Seção 5.1, normalizadas em relação à distância limite de medição $L = 14u.m.$.

A.1 Resultados para Comparação com Outros Algoritmos

Tabela 9: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.1, em cenários com as mesmas especificações utilizadas por (LANGENDOEN; REIJERS, 2003)

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,266361490	0,292970266	0,317900883	0,342235282	0,366384186
BSA-HTA	0,023886360	0,067014139	0,087950992	0,103492970	0,117223716
BSA-MMA	0,025748123	0,103094380	0,151249155	0,169925483	0,190958302
PSO-HTA	0,032260331	0,075427903	0,097177299	0,111582074	0,122656950
PSO-MMA	0,041191735	0,115582616	0,162604742	0,183526805	0,207007374

A.2 Resultados para Diferentes Número de Nós de Referência e Desconhecidos

Tabela 10: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 150 nós desconhecidos e 9 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,362926290	0,395674441	0,417007975	0,428093600	0,439671964
BSA-HTA	0,103949345	0,187269427	0,232578679	0,255603840	0,277327794
BSA-MMA	0,083471393	0,183910305	0,252513492	0,270961831	0,324195241
PSO-HTA	0,118230175	0,188970433	0,249711241	0,260420131	0,286574146
PSO-MMA	0,115250122	0,196704794	0,271077165	0,305357097	0,335407575

Tabela 11: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 150 nós desconhecidos e 16 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,224228218	0,238444210	0,247984790	0,260519324	0,272785389
BSA-HTA	0,066020124	0,120843878	0,140220290	0,154188876	0,170067963
BSA-MMA	0,017862462	0,095402072	0,125697470	0,152889174	0,182266371
PSO-HTA	0,069819971	0,124652056	0,145910065	0,160522141	0,176217572
PSO-MMA	0,033240566	0,107745723	0,139106620	0,161101683	0,185685390

Tabela 12: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 150 nós desconhecidos e 25 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,159413656	0,178537356	0,193564554	0,209473454	0,224951912
BSA-HTA	0,012386171	0,048231787	0,065108795	0,076586238	0,086950397
BSA-MMA	0,001452309	0,054509013	0,073150763	0,086949855	0,104334195
PSO-HTA	0,014172086	0,054122392	0,067078131	0,079724066	0,086696268
PSO-MMA	0,013477613	0,060057598	0,081903612	0,099226311	0,108829299

Tabela 13: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 250 nós desconhecidos e 9 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,236635056	0,279120309	0,313776429	0,346628670	0,378349510
BSA-HTA	0,022614415	0,062711560	0,083591963	0,094429113	0,104890854
BSA-MMA	0,029247983	0,117067579	0,163924483	0,184945497	0,195217834
PSO-HTA	0,029243511	0,063887672	0,091643844	0,104845685	0,116545745
PSO-MMA	0,040822874	0,123539405	0,178791888	0,202745679	0,206188057

Tabela 14: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 250 nós desconhecidos e 16 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,150982382	0,197308173	0,235172096	0,270704185	0,304874170
BSA-HTA	0,001974140	0,037763499	0,052022055	0,063420284	0,072766712
BSA-MMA	0,003994672	0,057698785	0,070312464	0,107238948	0,120610660
PSO-HTA	0,003755642	0,038150294	0,052260834	0,063727714	0,073142971
PSO-MMA	0,007506761	0,061060050	0,072000125	0,110557640	0,121798172

Tabela 15: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 250 nós desconhecidos e 25 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,108018139	0,144937347	0,179830904	0,213912905	0,247818452
BSA-HTA	0,000833171	0,031322812	0,053186519	0,062241642	0,069807690
BSA-MMA	0,000437676	0,037353003	0,052835974	0,068027230	0,078697911
PSO-HTA	0,002257064	0,033047404	0,054356666	0,063233410	0,070224789
PSO-MMA	0,004242507	0,039575247	0,053366942	0,070143014	0,080167443

Tabela 16: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 350 nós desconhecidos e 9 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,221998086	0,292990280	0,347211831	0,395529167	0,440439120
BSA-HTA	0,017286700	0,048997285	0,064094596	0,077054526	0,088131613
BSA-MMA	0,025423247	0,080759498	0,111747785	0,147334025	0,191926786
PSO-HTA	0,021699442	0,051280152	0,066108007	0,079458227	0,091261984
PSO-MMA	0,035726550	0,085184198	0,118141849	0,154221315	0,197693707

Tabela 17: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 350 nós desconhecidos e 16 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,135472964	0,205960111	0,262166984	0,312983855	0,360083490
BSA-HTA	0,001071853	0,028803816	0,040018109	0,048839032	0,056618676
BSA-MMA	0,003870074	0,043692178	0,067756781	0,094235233	0,108054626
PSO-HTA	0,000956721	0,028815704	0,040145525	0,049399393	0,057037941
PSO-MMA	0,004419529	0,044862756	0,070120613	0,096015615	0,108069849

Tabela 18: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.2, em cenários com 350 nós desconhecidos e 25 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,095014782	0,163261263	0,221719706	0,273805449	0,322270931
BSA-HTA	0,000021435	0,025266121	0,034941613	0,043273293	0,049179242
BSA-MMA	0,000296330	0,029247031	0,046718153	0,057529637	0,074196931
PSO-HTA	0,000024900	0,025747243	0,035707329	0,042770130	0,049246362
PSO-MMA	0,000399380	0,029235496	0,047340766	0,057168661	0,073805729

A.3 Resultados para Diferentes Topologias

Tabela 19: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.3, em enxames com topologia em malha quadrada cercada por referências

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,041056976	0,158380604	0,208066847	0,247033823	0,281950776
BSA-HTA	0,000000892	0,030326476	0,041830050	0,050606812	0,057792571
BSA-MMA	0,001148726	0,037688133	0,054082142	0,066658077	0,082024802
PSO-HTA	0,000002096	0,030326204	0,041830031	0,050606980	0,057792619
PSO-MMA	0,002549845	0,038074528	0,054261885	0,066701890	0,082079973

Tabela 20: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.3, em enxames com topologia em malha na forma de um C

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,052741876	0,168580097	0,220147063	0,263368479	0,301293946
BSA-HTA	0,000026829	0,031674233	0,043200084	0,051809783	0,058707277
BSA-MMA	0,002042917	0,042379876	0,061744237	0,073796354	0,077757375
PSO-HTA	0,000066204	0,031678613	0,043198110	0,051809915	0,058741912
PSO-MMA	0,001825539	0,042345449	0,062013596	0,076695230	0,080357688

Tabela 21: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.3, em enxames com ausência de topologia

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,144647303	0,165641461	0,184628347	0,204615811	0,225507157
BSA-HTA	0,011195974	0,052457508	0,067923620	0,081128226	0,099775078
BSA-MMA	0,004632224	0,067661751	0,088464815	0,115069792	0,127833328
PSO-HTA	0,012478594	0,052629719	0,067407445	0,081511686	0,099852004
PSO-MMA	0,006896296	0,070894038	0,091368505	0,118864401	0,131380729

Tabela 22: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.3, em enxames com topologia em malha quadrada com referências internas

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	0,739969882	0,821912055	0,857339963	0,886595310	0,913123964
BSA-HTA	0,000114397	0,037855439	0,060271512	0,093538911	0,119940842
BSA-MMA	0,003521342	0,039016162	0,056353124	0,070495013	0,085112463
PSO-HTA	0,192880661	0,248817906	0,269838995	0,283728285	0,302866224
PSO-MMA	0,185940461	0,232737871	0,262544642	0,301529686	0,328850759

A.4 Resultados com o Método MCMM

Tabela 23: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.4, em cenários com 150 nós desconhecidos e 4 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	1,859893497	1,865083564	1,866094017	1,881024442	1,885027755
MCMM	1,278017903	1,276126174	1,298527652	1,288381654	1,293188517
BSA-MMA	1,025193201	1,032032310	1,093645187	1,083742464	1,091127119

Tabela 24: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.4, em cenários com 250 nós desconhecidos e 4 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	1,510990918	1,512719627	1,513943335	1,515937222	1,516729527
MCMM	0,766167415	0,692310691	0,711806047	0,753351876	0,822761305
BSA-MMA	0,231005675	0,436953454	0,633820023	0,695618601	0,728654481

Tabela 25: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.4, em cenários com 350 nós desconhecidos e 4 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	1,450617524	1,452788463	1,454635686	1,456475398	1,458325467
MCMM	0,726883106	0,505868908	0,606835607	0,680766162	0,767689483
BSA-MMA	0,192891565	0,456497317	0,454481456	0,459261623	0,563322627

Tabela 26: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.4, em cenários com 500 nós desconhecidos e 6 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	1,789739768	1,791250984	1,792331624	1,793250715	1,795303194
MCMM	0,763350060	0,663377882	0,744395807	0,771361375	0,815967578
BSA-MMA	0,344515879	0,499320359	0,694327863	0,691487766	0,654213311

Tabela 27: $MMEPs$ normalizadas obtidas nas simulações da Seção 5.1.4, em cenários com 1000 nós desconhecidos e 9 nós de referência

	Variância da medida de distância				
Algoritmo	0,000	0,025	0,050	0,075	0,100
Min-Max	1,987500200	1,990279904	1,991161350	1,992218141	1,993157734
MCMM	0,713265915	0,711380841	0,762061217	0,872935109	0,955115924
BSA-MMA	0,409182952	0,623312968	0,766968373	0,811651065	0,836542418

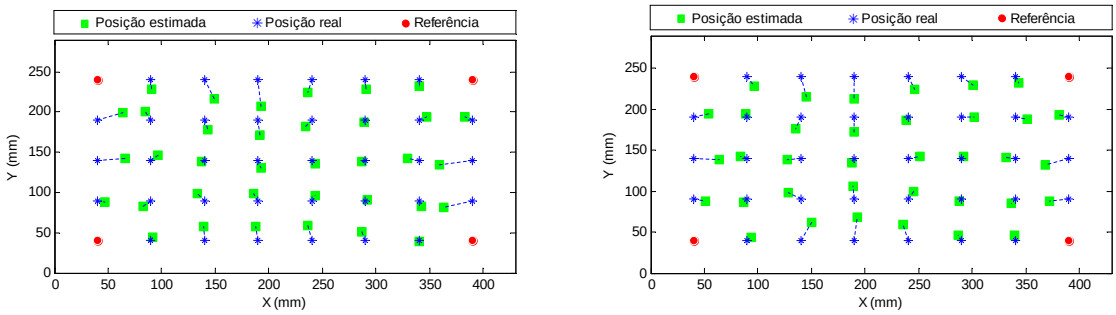
APÊNDICE B – Resultados de Experimentos

Este apêndice apresenta os valores das $MMEPs$ obtidas nos experimentos da Seção 5.2, bem como exemplos de mapas de localização resultantes de cada tipo de experimento. Para descrever a evolução do processo de localização, são apresentadas as $MMEPs$ alcançadas ao fim das ETAPAS-II e III, bem como as $MMEPs$ obtidas nas iterações 2, 5, 10, 20 e 40 da ETAPA-IV dos algoritmos BSA-HTA e BSA-MMA.

B.1 Resultados para Diferentes Tamanhos de Enxame

Tabela 28: $MMEPs$ (mm) obtidas nos experimentos da Seção 5.2.2, em enxames com 36 nós desconhecidos e 4 nós de referência

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	30,538	30,538	15,442	13,500	13,088	12,823	12,809
BSA-MMA	30,559	30,559	13,238	13,572	13,320	13,251	13,318



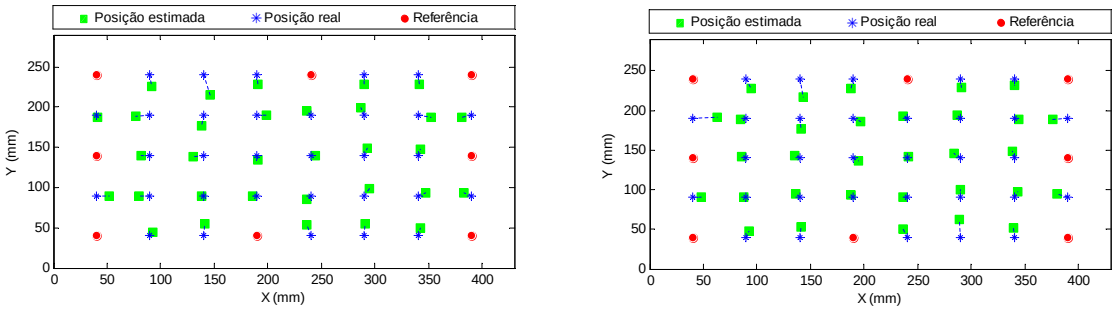
(a) BSA-HTA

(b) BSA-MMA

Figura 57: Exemplos do resultado final do processo de localização em enxames com 36 nós desconhecidos e 4 nós de referência

Tabela 29: $MMEPs$ (mm) obtidas nos experimentos da Seção 5.2.2, em enxames com 32 nós desconhecidos e 8 nós de referência

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	19,821	19,821	10,601	10,055	10,021	9,886	9,885
BSA-MMA	19,745	19,745	11,962	9,606	9,545	9,587	9,544

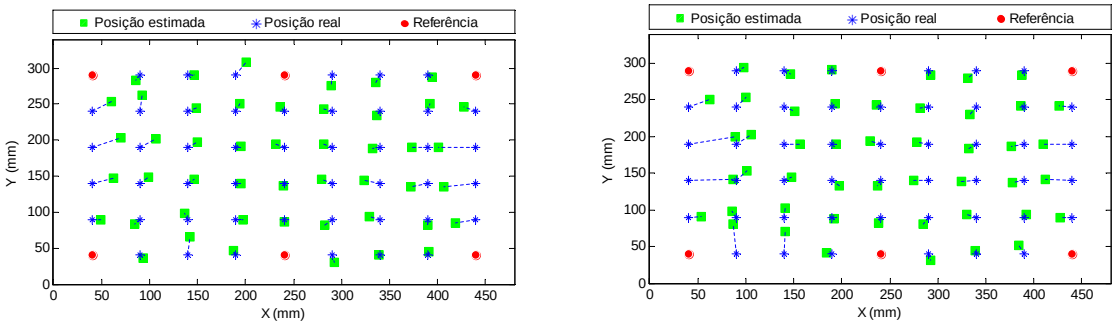


(a) BSA-HTA (b) BSA-MMA

Figura 58: Exemplos do resultado final do processo de localização em enxames com 32 nós desconhecidos e 8 nós de referência

Tabela 30: $MMEPs$ (mm) obtidas nos experimentos da Seção 5.2.2, em enxames com 48 nós desconhecidos e 6 nós de referência

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	24,150	24,150	14,689	13,274	13,047	12,802	12,906
BSA-MMA	22,735	22,735	15,542	15,322	15,004	14,904	14,957

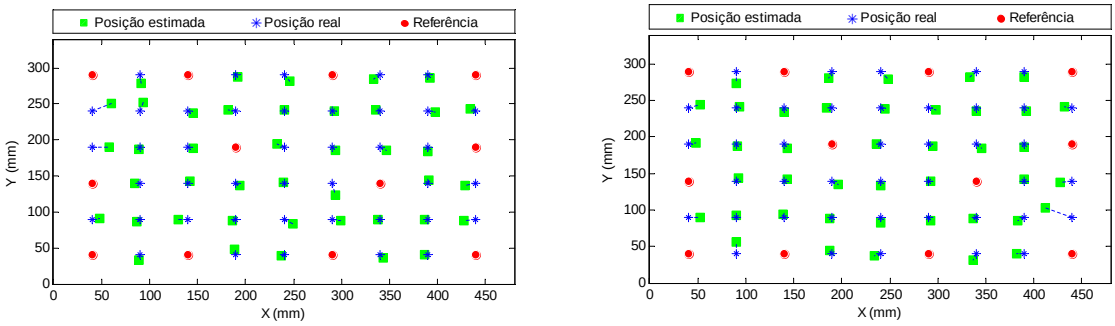


(a) BSA-HTA (b) BSA-MMA

Figura 59: Exemplos do resultado final do processo de localização em enxames com 48 nós desconhecidos e 6 nós de referência

Tabela 31: $MMEPs$ (mm) obtidas nos experimentos da Seção 5.2.2, em enxames com 42 nós desconhecidos e 12 nós de referência

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	20,694	20,694	9,046	7,715	7,656	7,468	7,758
BSA-MMA	20,688	20,688	8,736	7,824	7,861	7,918	8,064

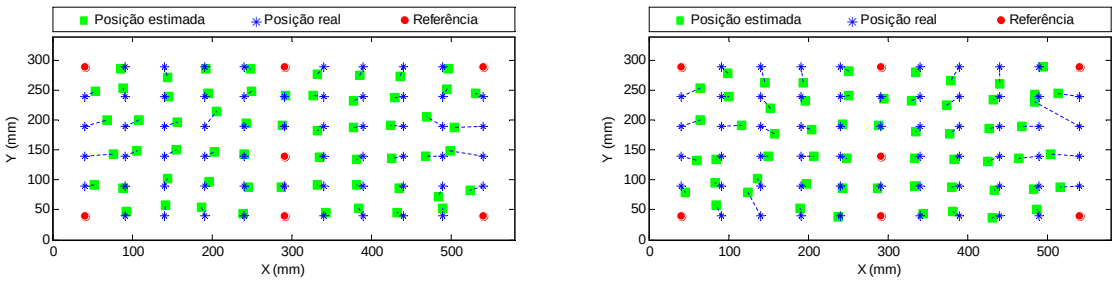


(a) BSA-HTA (b) BSA-MMA

Figura 60: Exemplos do resultado final do processo de localização em enxames com 42 nós desconhecidos e 12 nós de referência

Tabela 32: $MMEPs$ (mm) obtidas nos experimentos da Seção 5.2.2, em enxames com 59 nós desconhecidos e 7 nós de referência

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	25,655	25,655	15,772	14,112	13,886	13,726	13,808
BSA-MMA	27,262	27,262	18,760	16,769	16,780	16,604	16,816

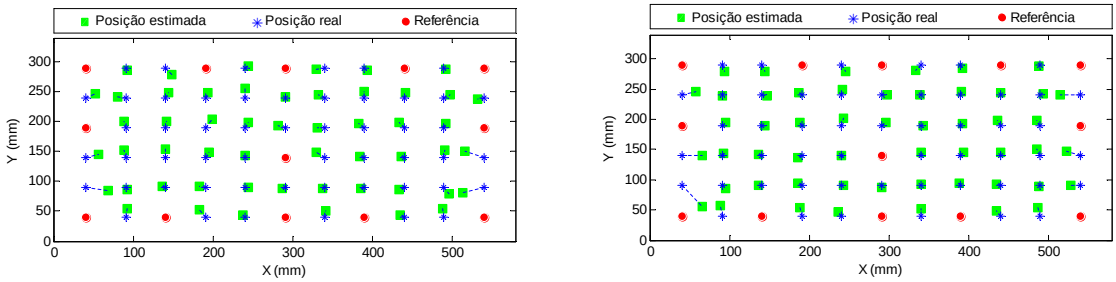


(a) BSA-HTA (b) BSA-MMA

Figura 61: Exemplos do resultado final do processo de localização em enxames com 59 nós desconhecidos e 7 nós de referência

Tabela 33: *MMEPs* (*mm*) obtidas nos experimentos da Seção 5.2.2, em enxames com 53 nós desconhecidos e 13 nós de referência

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	20,765	20,765	11,472	10,133	9,764	9,801	10,142
BSA-MMA	19,980	19,980	10,314	9,343	9,354	9,391	9,578



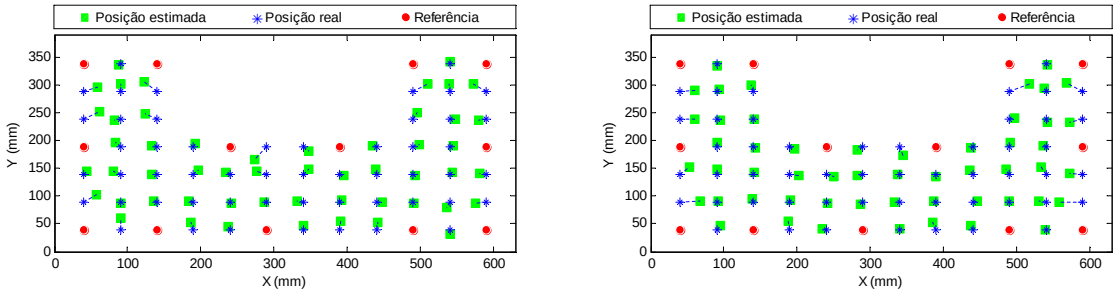
(a) BSA-HTA (b) BSA-MMA

Figura 62: Exemplos do resultado final do processo de localização em enxames com 53 nós desconhecidos e 13 nós de referência

B.2 Resultados para Diferentes Topologias

Tabela 34: *MMEPs* (*mm*) obtidas nos experimentos da Seção 5.2.3, em enxames com topologia em malha na forma de um C

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	24,563	24,563	11,857	10,704	10,334	10,353	10,764
BSA-MMA	20,255	20,255	10,001	9,215	9,094	9,113	9,63

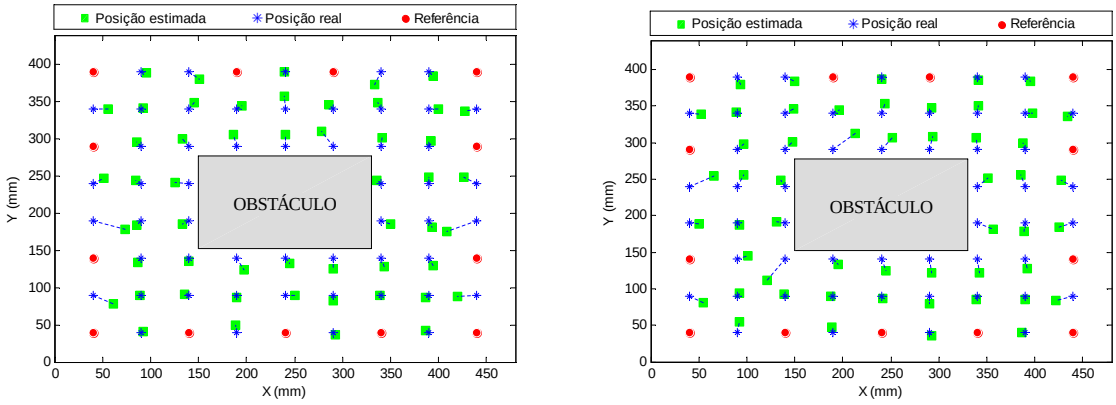


(a) BSA-HTA (b) BSA-MMA

Figura 63: Exemplos do resultado final do processo de localização em enxames com topologia em malha na forma de um C

Tabela 35: *MMEPs* (*mm*) obtidas nos experimentos da Seção 5.2.3, em enxames com topologia em malha com obstáculo

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	21,873	21,873	12,878	10,562	10,560	10,720	10,910
BSA-MMA	23,508	23,508	13,652	12,032	11,906	12,152	12,308

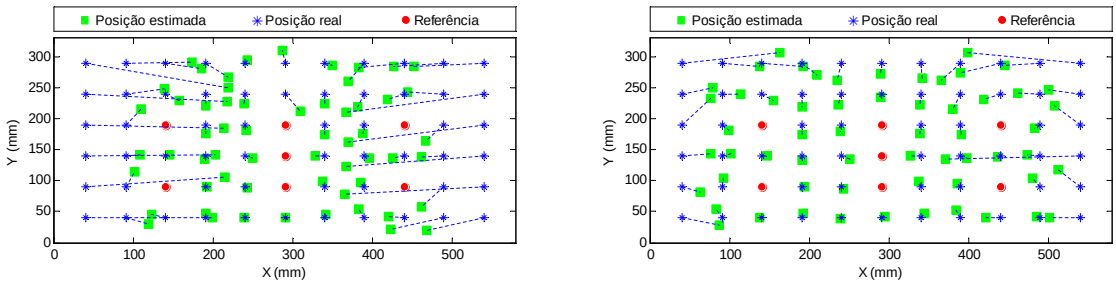


(a) BSA-HTA (b) BSA-MMA

Figura 64: Exemplos do resultado final do processo de localização em enxames com topologia em malha com obstáculo

Tabela 36: *MMEPs* (*mm*) obtidas nos experimentos da Seção 5.2.3, em enxames com topologia em malha retangular com referências internas

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	77,029	77,029	49,458	42,978	48,276	49,807	50,390
BSA-MMA	73,139	73,139	36,003	43,482	32,054	32,456	32,052

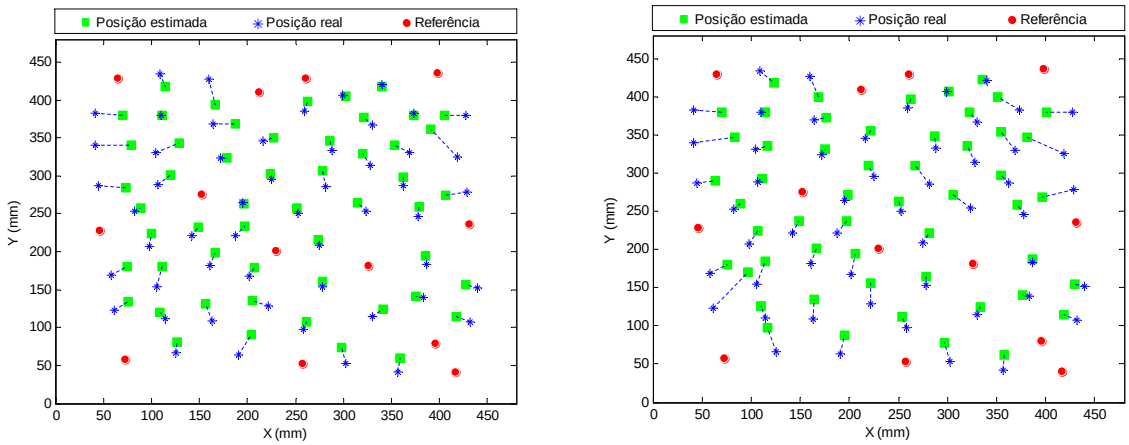


(a) BSA-HTA (b) BSA-MMA

Figura 65: Exemplos do resultado final do processo de localização em enxames com topologia em malha retangular com referências internas

Tabela 37: *MMEPs* (*mm*) obtidas nos experimentos da Seção 5.2.3, em enxames com ausência de topologia

Algoritmo	ETAPA-II (Min-Max)	ETAPA-III (MCMM)	Iterações da ETAPA-IV				
			2	5	10	20	40
BSA-HTA	32,378	32,378	17,903	17,585	17,059	17,138	17,199
BSA-MMA	30,199	30,199	20,619	19,482	19,224	19,164	19,590



(a) BSA-HTA (b) BSA-MMA

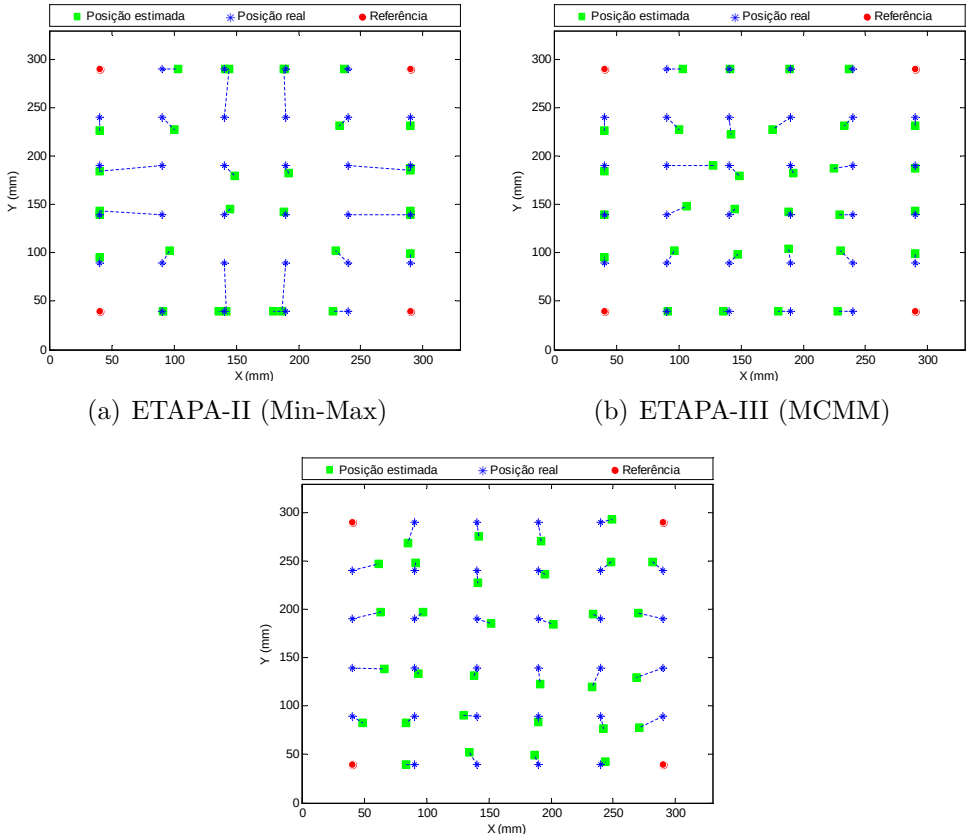
Figura 66: Exemplos do resultado final do processo de localização em enxames com ausência de topologia

B.3 Resultados com o Método MCMM

Tabela 38: *MMEPs* (*mm*) obtidas nos experimentos da Seção 5.2.4, em experimentos com o método MCMM

Número de nós		ETAPA-II	ETAPA-III	Iterações da ETAPA-IV (BSA-MMA)				
D(*)	R(**)	(Min-Max)	(MCMM)	2	5	10	20	40
32	4	18,075	11,550	15,008	13,507	13,117	13,257	13,296
60	6	18,140	13,045	16,843	14,874	14,685	14,813	15,087

(*) Desconhecidos (**) Referências

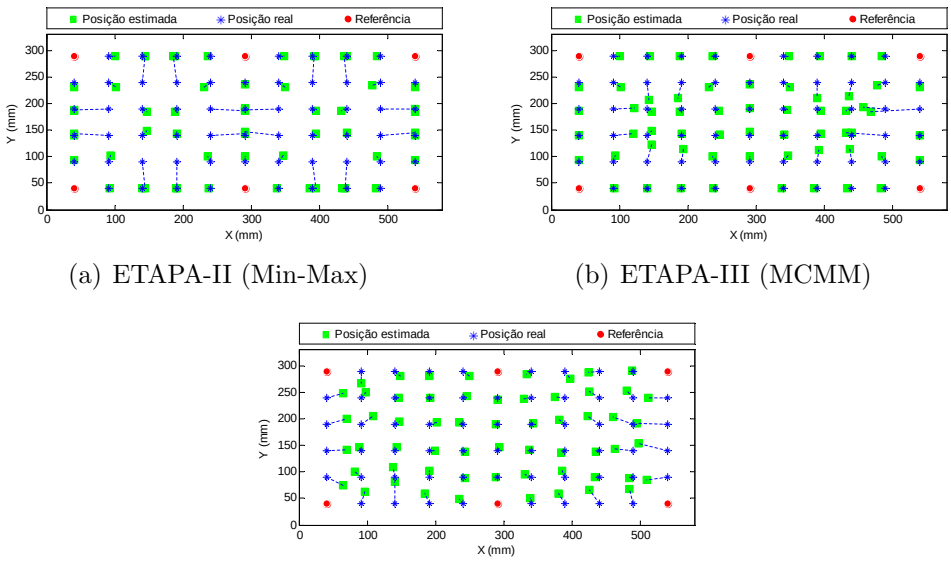


(a) ETAPA-II (Min-Max)

(b) ETAPA-III (MCMM)

(c) ETAPA-IV (BSA-MMA)

Figura 67: Evolução do processo de localização em enxame com 32 nós desconhecidos e 4 nós de referência, considerando restrições quanto à divulgação dos nós de referência



(a) ETAPA-II (Min-Max)

(b) ETAPA-III (MCMM)

(c) ETAPA-IV (BSA-MMA)

Figura 68: Evolução do processo de localização em enxame com 60 nós desconhecidos e 6 nós de referência, considerando restrições quanto à divulgação dos nós de referência