Um Protocolo de Localização Assíncrono para RSSF Heterogêneas Centradas em Ator

José Souza de Jesus, Paulo André da S. Gonçalves

Centro de Informática (CIn)
Universidade Federal de Pernambuco (UFPE)
50.740-560 – Recife – PE – Brasil

{jsj, pasg}@cin.ufpe.br

Resumo. *Nas RSSF heterogêneas centradas em ator, o ator é capaz de organizar sensores próximos com ciclos de trabalho distintos e formar uma rede de curta duração para fins específicos. Nessas redes, os sensores se localizam em relação ao ator através do uso de protocolos assíncronos de treinamento de localização. Na literatura, existem dois protocolos propostos para isso: Flat+ e Binary Training. Contudo, tais protocolos não consideram os impactos de imperfeições no canal de comunicação. Essas imperfeições levam a problemas de convergência de treinamento e a um aumento nos erros de localização. Este trabalho analisa o desempenho desses protocolos em canais com desvanecimento e propõe um protocolo assíncrono de treinamento de localização que lida melhor com imperfeições no canal. As avaliações de desempenho do protocolo proposto mostram que o mesmo não apresenta problemas de convergência e melhora a acurácia de localização em relação aos protocolos estudados.*

Abstract. *In actor-centric heterogeneous WSNs, the actor is able to organize nearby sensors running different dutty-cicle schemes in order to form a short-lived network for specific objectives. In such networks, the sensors use asynchronous location training protocols to discover their location. There are two asynchronous protocols proposed to perform location training: Flat+ e Binary Training. However, both protocols do not consider the impacts imperfections on the communication channel. These imperfections leads to convergence problems and an increase in localization errors. This work evaluates the performance of both protocols under fading channels and proposes an asynchronous location protocol that deals better with channel imperfections. Performance evaluation shows that the proposed protocol does not have convergence problems and improves the localization accuracy with respect to the protocols studied.*

1. Introdução

Em redes de sensores e atores, um dispositivo ator é capaz de organizar os sensores que estão em sua proximidade em uma rede de curta duração e centrada no próprio ator [Barsi et al. 2011]. O objetivo principal é a realização de tarefas específicas definidas pelo ator. Após a conclusão dessas tarefas, os sensores voltam ao estado desorganizado, sendo desfeita a rede criada pelo ator. Em geral, e ao contrário dos atores, os sensores podem ter restrições mais significativas de processamento, memória, energia e potência de transmissão. A quantidade de sensores utilizados em uma aplicação pode exceder os milhares e os mesmos podem estar espalhados por uma vasta área geográfica.

Dependendo da aplicação, os dados coletados pelos sensores só possuem utilidade caso venham com alguma informação de localização do ponto de coleta. A informação de localização pode se referir, por exemplo, ao posicionamento do sensor de forma absoluta (*e.g.* coordenadas geográficas) ou relativa a um ponto [Yick et al. 2008]. Para possibilitar que um sensor descubra sua localização, diversas técnicas foram propostas na literatura [Mao et al. 2007]. As mais comuns consistem em estimativa de distância, utilização de nós âncora com localização conhecida e algoritmos para refinamento (multilateração, por exemplo). A grande maioria depende da comunicação entre os nós durante o processo de localização.

Recentemente, foi introduzido o conceito de treinamento de localização através da execução de protocolos assíncronos [Barsi et al. 2009, Barsi et al. 2011]. O treinamento de localização consiste no nó sensor determinar sua posição apenas escutando *beacons* transmitidos pelo ator. Esse tipo de abordagem é mais escalável por dispensar a comunicação entre os sensores durante o processo de localização. Na literatura, dois protocolos assíncronos são capazes de realizar o treinamento em redes heterogêneas, onde cada sensor obedece a seu próprio ciclo de trabalho: o *Flat+* [Barsi et al. 2009] e o *Binary Training* [Barsi et al. 2011]. Entretanto, ambos protocolos foram desenvolvidos considerando-se apenas o uso sob canais de comunicação ideais.

Este trabalho considera uma rede de sensores sem fio heterogênea e centrada em ator espalhada num ambiente externo como áreas rurais. O canal de comunicação é modelado com desvanecimento e permite a ocorrência de perda de pacotes. O desempenho do *Flat+* e do *Binary Training* é estudado no cenário descrito. As avaliações mostram que as imperfeições no canal de comunicação levam eles a problemas de convergência de treinamento e resultam em um aumento nos erros de localização. A partir disso, este trabalho propõe um protocolo assíncrono de localização mais resistente aos impactos causados por imperfeições no canal de comunicação. Além disso, o protocolo proposto oferece a possibilidade de melhorar, de forma configurável, a precisão na localização do sensores de acordo com os requisitos da aplicação.

Este artigo está organizado como segue: a Seção 2 apresenta os trabalhos relacionados. A Seção 3 define o modelo de redes de sensores estudado neste trabalho. A Seção 4 descreve os protocolos *Flat+* e *Binary Training*. A Seção 5 apresenta uma avaliação do desempenho desses protocolos. A Seção 6 apresenta o protocolo proposto neste trabalho. A Seção 7 apresenta uma avaliação de desempenho do protocolo proposto, comparando-os com os demais protocolos estudados. Por fim, a Seção 8 apresenta as conclusões.

2. Trabalhos Relacionados

Existem diversos estudos relacionados ao problema de localização em redes de sensores sem fio [Mao et al. 2007, Yick et al. 2008]. Os algoritmos de localização existentes podem ser classificados em duas categorias [He et al. 2005]: *range-based* e *range-free*. Os algoritmos do tipo *range-based* permitem que o sensor estime sua distância Euclidiana até um ponto de referência (*e.g* um dispositivo âncora), derivando sua localização através da combinação dessas informações. Os algoritmos do tipo *range-free* dispensam a medida da distância Euclidiana, geralmente considerando sistemas de coordenadas discretas. Os algoritmos *range-free* geralmente permitem um controle sobre o nível de granularidade do erro, embora produzam resultados menos precisos se comparados às abordagens *range-*

based. Por outro lado, as soluções *range-based* são mais suscetíveis às condições do meio de comunicação [He et al. 2005].

A maioria das propostas *range-based* estimam a distância através da força do sinal recebido ou RSSI (*Received Signal Strength Indicator*) [Mao et al. 2007, Cassano et al. 2009]. Em [Lin et al. 2009] é avaliado o efeito da perda de *beacons* na eficiência de localização via RSSI num ambiente interno real, especificamente utilizando um algoritmo KNN (*k-Nearest Neighbor*). Na avaliação, é constatada uma diferença significativa na eficiência ao se considerar canais com ruído. O maior problema do uso do RSSI está no erro inserido pela estimativa da distância, uma vez que medidas de RSSI apresentam variabilidade devido às condições do canal de comunicação [Patwari and Kasera 2011].

As abordagens *range-free* geralmente modelam o espaço discretamente. O algoritmo *Centroid* [Bulusu et al. 2000] permite que nós se localizem calculando o centroide do polígono formado por âncoras através da transmissão de *beacons*. A eficiência do algoritmo depende da distribuição das âncoras. Outra possibilidade é obter uma localização aproximada através da distância em saltos (*hops*) [Nagpal et al. 2003]. Em [Wang and Zhu 2009] é proposto um mecanismo de localização que estima a distribuição do posicionamento dos nós através de um método de *Monte Carlo*. Por se tratar de um modelo probabilístico, esse método pode ser aplicado em redes móveis e em ambientes com rádio de comportamento irregular. A eficiência do algoritmo depende da densidade de âncoras.

Recentemente, a ideia de treinamento de localização para redes de sensores e atores foi introduzida na literatura [Barsi et al. 2009, Barsi et al. 2011]. Esse tipo de treinamento é um método *range-free*, sendo vantajoso em relação a outros métodos por dispensar o uso de múltiplas âncoras. Em [Barsi et al. 2009, Barsi et al. 2011] é considerado um esquema de localização baseado em setores formados por coroas circulares com o ator ao centro como origem de um sistema de coordenadas polares discretas. Os protocolos dispensam a sincronização entre os sensores e a comunicação entre eles durante o processo de treinamento, permitindo uma alta escalabilidade. Este trabalho realiza uma avaliação de protocolos assíncronos de treinamento num cenário mais realista, além de propor de um novo protocolo que melhor lida com perda de pacotes. Essas são as duas maiores contribuições deste trabalho.

3. Modelo de Rede

Este trabalho considera uma rede de sensores composta por sensores não sincronizados e um ator. Cada sensor n, na rede formada pelo ator, deve estimar sua própria localização em relação ao ator. Para isso, o sensor se apoia apenas na escuta de *beacons* transmitidos pelo ator. O treinamento dispensa o emprego de múltiplas âncoras, não exige comunicação adicional e dispensa sincronização entre os sensores, ocasionando em alta escalabilidade.

A seguir, são apresentadas algumas considerações sobre o modelo de rede adotado. Tais considerações também são feitas nos estudos em [Barsi et al. 2009] e [Barsi et al. 2011]. Para a realização do treinamento, é admitido que o ator seja capaz de transmitir *beacons* em níveis de potência diferentes através de uma antena isotrópica, resultando em coroas circulares delimitadas pelo alcance de transmissão em cada nível.

A Figura 1 ilustra uma rede centrada num ator e cujo treinamento de localização resulta em 5 coroas ($C_0 \ldots C_4$) possíveis para o posicionamento dos sensores compondo a rede formada pelo ator. Os protocolos estudados aqui apenas lidam com o treinamento dos sensores em relação às coroas. Contudo, é possivel realizar um treinamento adicional de localização em setores através de transmissões por parte do ator com uma antena direcional [Barsi et al. 2011].

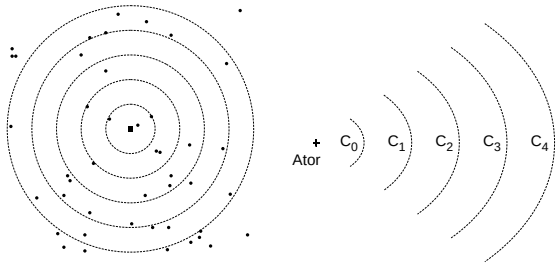
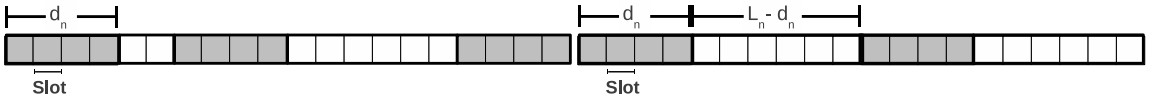


Figura 1. Rede centrada em ator divida em 5 coroas.

O canal de comunicação é dividido em *slots* de tempo de tamanho fixo e a rede é composta por dois tipos de sensores: periódicos e livres. A Figura 2 ilustra o ciclo de trabalho de cada tipo de sensor. Cada sensor n obedece a um ciclo de trabalho próprio, onde escuta por um período de d_n *slots* e entra em inatividade até o próximo período de escuta. No caso do sensor periódico, a soma do tempo de atividade (d_n) com o tempo de inatividade é sempre fixa e igual a L_n. Já o sensor livre é capaz de alterar dinamicamente seu período de inatividade ao longo do tempo.

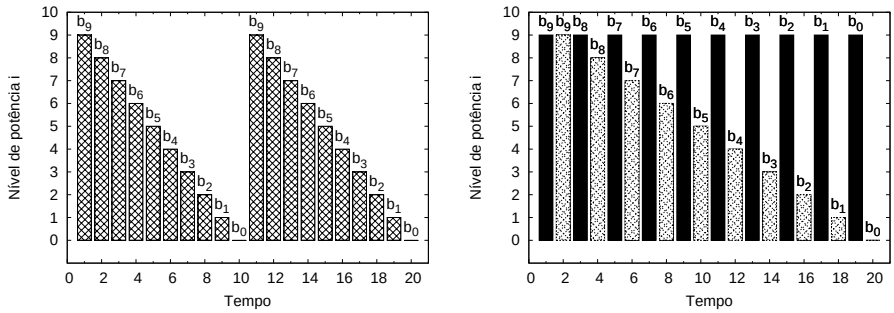


(a) Sensor livre (b) Sensor periódico

Figura 2. Tipos de sensores na rede.

4. Protocolos Assíncronos de Treinamento

Esta seção detalha os protocolos *Flat+* [Barsi et al. 2009] e *Binary Training* [Barsi et al. 2011]. As únicas condições necessárias para a execução desses protocolos são: os sensores conhecerem o comportamento do ator e serem capazes de se sincronizar com o ator toda vez que ouvem um *beacon*. O comportamento do ator para o protocolo *Flat+* é ilustrado na Figura 3(a). O ator transmite um cliclo de k *beacons*, onde cada *beacon* está em um *slot* distinto. A cada ciclo de transmissão de *beacons*, o nível de potência parte sequencialmente do maior nível para o menor nível. A Figura 3(b) ilustra o comportamento do ator com o protocolo *Binary Training*. Pares de *beacons* sucessivos b_i são transmitidos em *slots* consecutivos. O primeiro *beacon* do par é transmitido com potência máxima e é chamado de *beacon control*. O segundo *beacon* do par é transmitido com potência de nível i e é conhecido por *beacon data*. À medida que os pares de *beacons* vão sendo transmitidos, a potência de transmissão dos *beacons data* é reduzida sequencialmente até o menor nível.



(a) *Flat* (b) *Binary*

Figura 3. Comportamento do ator para 10 coroas.

4.1. O protocolo *Flat+*

O protocolo *Flat+* mantém um vetor $R_k[\,]$ que armazena a informação se um *beacon* é ou não recebido pelo sensor. O sensor estará treinado na coroa C_i, $\forall i > 0$, se receber o *beacon* b_i e não receber b_{i-1}. Para o caso especial $i = 0$, o sensor estará treinado na coroa C_0 se receber o *beacon* b_0. O Algoritmo 4.1 detalha como os sensores realizam o treinamento. Os registradores min e max armazenam, respectivamente, o menor *beacon* ouvido e o maior não ouvido. Após a recepção do primeiro *beacon*, o sensor se sincroniza com o ator, determina via *backtracking* quais *beacons* previamente transmitidos não foram escutados (linhas 9-11), e ajusta min e max (linhas 12 e 18). Esta sincronização também permite ao sensor determinar qualquer *beacon* não escutado (linha 22).

Para uma maior eficiência, o protocolo assume que se o sensor recebe um *beacon* b_i então receberá qualquer *beacon* b_j, $j > i$. De modo análogo, se não recebe b_i, então não receberá qualquer b_j, $j < i$. Logo, como o comportamento do ator é conhecido, é possível prever e evitar ciclos L_n onde a escuta se torna desnecessária. Esse é o princípio do procedimento $Wait()$ (linha 34), o qual ignora o próximo ciclo L_n se $\forall b_i$ a ser transmitido em L_n tem-se $R_k[i] \neq 0$.

Note que o *Flat+* treina apenas sensores periódicos. O treinamento é possível caso $d_n \geq mdc(L_n, k)$. No pior caso, são utilizados $v_{max} = \frac{k}{mdc(L_n,k)}$ ciclos, o tempo de atividade do sensor é de $\omega_{max} = d_n \cdot v_{max}$ e o tempo total decorrido é $\tau_{max} = L_n \cdot v_{max}$ [Barsi et al. 2009]. A função $mdc(x, y)$ indica o máximo divisor comum de x e y.

4.2. O protocolo *Binary Training*

O protocolo *Binary Training* [Barsi et al. 2011] possui algumas vantagens em relação ao *Flat+*, como melhor desempenho e maior resistência a desvios nos relógios dos dispositivos. O Algoritmo 4.2 detalha como é realizado o treinamento pelos sensores. O protocolo utiliza os registradores min e max para determinar a coroa C_i, $0 \leq i < k$, na qual o sensor está localizado. A coroa C_i estimada estará no intervalo $[min, max]$. A proposta utiliza busca binária para determinar o *beacon* de interesse b_{guess}, onde $guess = \lceil \frac{min+max}{2} \rceil$, e reduzir o intervalo $[min, max]$.

Os *beacons* são analisados a cada par de *slots* (linhas 6-9 e 11-27) para determinar qual deles foi *control* e *data*. max é atualizado quando o sensor ouve um *control* mas não ouve o seu respectivo *data* (linhas 16-19 ou 20-23). Por sua vez, min é atualizado

Algoritmo 4.1 *Flat+*

```
1:  ouviu ← falso
2:  treinado ← falso
3:  min ← k − 1; max ← 0; v ← 0
4:  inicializa cada elemento de R_k[ ] ← 0
5:  enquanto ¬treinado faça
6:      v ← v + 1
7:      para i ← 0 até d_n − 1 faça
8:          se recebeu b_c então
9:              se ¬ouviu então
10:                 ouviu ← verdade
11:                 sincroniza e realiza backtracking
12:                 max ← maior beacon não ouvido
13:             para h ← c até min faça
14:                 R_k[h] ← 1
15:             se c = 0 ou (R_k[c] = 1 e R[c − 1] = −1) então
16:                 treinado ← verdade
17:                 coroa ← C_c
18:             min ← c
19:             t ← t + 1
20:         senão
21:             se ouviu então
22:                 c ← k − 1 − t mod k
23:                 para h ← max até c faça
24:                     R_k[h] ← −1
25:                 max ← c
26:                 se R_k[c + 1] = 1 então
27:                     treinado ← verdade
28:                     coroa ← C_{c+1}
29:                 t ← t + 1
30:         se ¬treinado então
31:             se ¬ouviu então
32:                 espera L_n − d_n slots
33:             senão
34:                 espera Wait() slots
```

Algoritmo 4.2 *Binary Training*

```
1:  treinado ← falso
2:  min ← −1; max ← k − 1; v ← 0; t ← 0
3:  enquanto ¬treinado faça
4:      para i ← 0 até d_n − 1 faça
5:          se i é par então
6:              se recebe b_c então
7:                  ini ← c
8:              senão
9:                  ini ← k
10:         senão
11:             se não recebe b_c então
12:                 se min ≤ ini então
13:                     min ← ini
14:                     controle ← t + i − 1
15:             senão
16:                 se c = ini então
17:                     se max ≥ c então
18:                         max ← c
19:                         controle ← t + i − 1
20:                 se ini ≠ k e c = (ini − 1) mod k então
21:                     se max ≥ ini então
22:                         max ← c
23:                         controle ← t + i
24:                 se ini = k então
25:                     se min ≤ (c + 1) mod k então
26:                         min ← (c + 1) mod k
27:                         controle ← t + i
28:     t ← t + d_n
29: se max − min = 1 então
30:     treinado ← verdade
31:     coroa ← C_max
32: senão
33:     interesse = ⌈(min+max)/2⌉
34:     espere por Wait() slots
```

pela escuta do par *control* e *data* (linhas 11-14 ou 24-27). O objetivo da busca binária é ativar o sensor apenas para escutar o *beacon* de interesse, a fim de reduzir o tamanho do intervalo $[min, max]$. Quando satisfeita a condição $max - min = 1$, o sensor estará treinado e pertencerá à coroa max (C_{max}). Caso contrário, o sensor entrará em inatividade, através da execução do $Wait()$, por um período determinado de *slots*. O procedimento $Wait()$ também calcula quantos *slots* são necessários para a transmissão do *beacon* b_{guess} e depende do tipo do sensor. Sensores livres calculam apenas o tempo necessário para a próxima transmissão de b_{guess}. No caso dos sensores periódicos, $Wait()$ ignora o número necessário de *slots* até o ciclo mais próximo onde b_{guess} será transmitido. Para ambos os tipos de sensores, o treinamento é possível se d_n for par. No caso dos sensores periódicos, também deve ser satisfeita a condição $\frac{d_n}{2} \geq mdc(\frac{L_n}{2}, k)$.

5. Avaliação de Desempenho do *Flat+* e do *Binary Training*

Os protocolos *Flat+* e *Binary Training* foram projetados considerando um canal de comunicação ideal livre de erros [Barsi et al. 2009, Barsi et al. 2011]. Nesses protocolos, a não recepção de um *beacon* b_i se traduz erroneamente na impossibilidade do sensor recebê-lo em um novo ciclo de transmissão do ator. Com isso, espera-se que esses protocolos sejam afetados negativamente em cenários com possibilidade de perdas de *beacons*. Em situações reais, a recepção de *beacons* depende de outros fatores relacionados ao

meio. Assim sendo, esta seção apresenta uma avaliação de desempenho desses protocolos com um canal de comuninicação imperfeito, sujeito à perda de *beacons* devido a efeitos de desvanecimento. Para as avaliações de desempenho dos protocolos foi desenvolvido um simulador em C++. A seguir são detalhados o modelo do canal de comunicação, os parâmetros de simulação, as métricas de avaliação e os resultados obtidos.

5.1. Modelagem do Canal de Comunicação

Neste trabalho, é considerado um canal de comunicação cujos sinais transmitidos sofrem variações por desvanecimento. Um modelo matemático utilizado na representação desse fenômeno é o modelo *log-distance path loss* [Wang and Berger 2008] definido por:

$$P_r = P_t - P_L(D_0) - 10 \cdot \alpha \cdot \log(\frac{D}{D_0}) + X_\sigma \quad , \tag{1}$$

onde P_t é a potência de transmissão, P_r é a potência do sinal recebido em dBm, D é a distância entre o transmissor e o receptor, e α é um expoente de atenuação que relaciona a perda de potência do sinal com a distância. X_σ é uma variável aleatória de distribuição normal com média em zero e desvio padrão σ e representa uma variação na força do sinal recebido. $P_L(D_0)$ é a atenuação verificada a uma distância de referência D_0 e é calculado através da expressão $20 \log \frac{4\pi}{\lambda}$. Geralmente, $D_0 = 1\ m$, o qual é adotado neste trabalho.

Dada uma potência mínima γ necessária para o receptor decodificar o sinal, a probabilidade de recepção com sucesso é calculada segundo a probabilidade do sinal médio recebido exceder γ. Logo, a probabilidade de recepção do sinal, P_{suc}, é dada por

$$P_{suc} = Q(\frac{\gamma - \overline{P_r}}{\sigma}) \quad , \tag{2}$$

onde $Q(x)$ é a função-Q da distribuição normal e $\overline{P_r}$ é o valor médio da potência recebida igual a $P_t - P_L(D_0) - 10 \cdot \alpha \cdot \log(\frac{D}{D_0})$.

Este trabalho considera propriedades de recepção de sensores com rádio *Chipcom CC2420*[1] Assim, é considerado um limiar de recepção $\gamma = -90\ dBm$ e uma modulação OQPSK (*offset quadrature phase-shift keying*) com codificação de 2 *bits* por símbolo. A taxa de erro de bits (BER) é calculada como $BER = 1 - \sqrt{P_{suc}}$. A taxa de erro de pacotes (*beacons*), PER, é dada por $PER = 1 - (1 - BER)^n$, onde n é o tamanho do pacote. Os experimentos conduzidos assumem *beacons* de 128 *bits* e é compatível com a especificação *IEEE 802.15.4* [IEEE Standard 2001].

O foco deste trabalho está em cenários externos abertos (sem obstruções de grande porte como edifícios), como áreas rurais, campos e pequenas cidades. O modelo *log-distance path loss* adotado, é amplamente utilizado para se estudar a propagação de sinais em ambientes externos [Wang et al. 2004, Laselva et al. 2005, Wang and Berger 2008]. Nesse caso, geralmente, é adotado $2 \leq \alpha \leq 4$ [Wang and Berger 2008], sendo aqui considerado $\alpha = 3$. O valor de σ nas avaliações deste artigo é igual a $2\ dBm$. A escolha deste valor é baseada em observações empíricas obtidas em [Wang et al. 2004, Laselva et al. 2005] para os tipos de cenários descritos.

[1]Chipcom (2004). CC2420 Data Sheet.

5.2. Resultados

Esta seção estuda o comportamento do *Flat+* e do *Binary Training* nos cenários definidos. As métricas de avaliação de desempenho utilizadas são as seguintes:

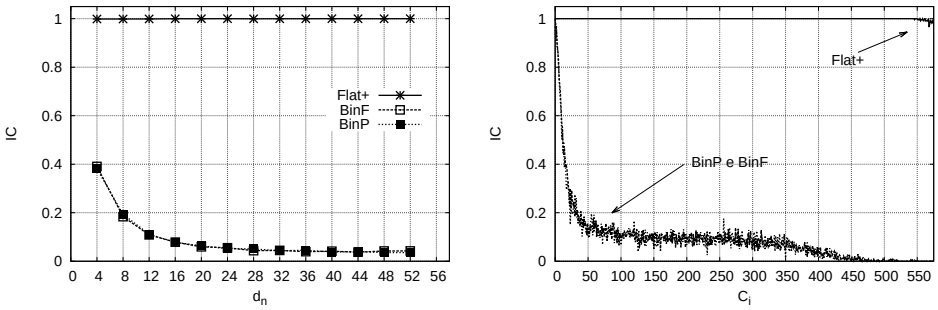
- **Erro (Δ)**: A diferença entre o nº da coroa estimada pelo sensor e o nº da coroa real na qual ele está posicionado;
- **Índice de Convergência (IC)**: a razão do número de sensores treinados, dado que o algoritmo terminou normalmente, sobre o número total de sensores.

As simulações consideram um cenário onde os sensores estão dispostos em um área circular com raio de $5.750\ m$ e o ator está localizado no centro da mesma. O espaço é dividido em 575 coroas ($k = 575$) de raio fixo de $10\ m$. O parâmetro k determina a quantidade de níveis de potência usada pelo ator e depende exclusivamente do rádio que ele utiliza. Os *slots* de tempo possuem duração de $2\ ms$. O Índice de Convergência e o Erro são avaliados em função do parâmetro d_n. Para cada valor de d_n são realizados 10.000 treinamentos de sensores cujas posições são escolhidas aleatoriamente na área circular. Os valores de d_n estudados estão no intervalo $[4, 8, .., 52]$. O valor de L_n é fixo em 54 *slots*. Os valores utilizados são os mesmos adotados em [Barsi et al. 2011], permitindo uma comparação direta com os resultados apresentados em tal trabalho.

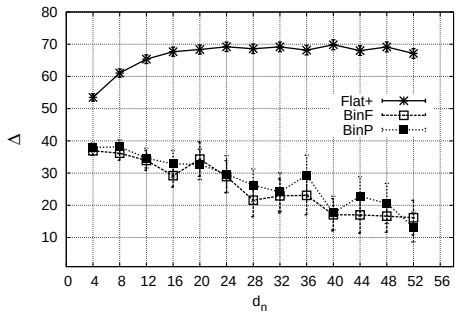
Para melhor apresentação foi adotada a convenção *BinF* para *Binary Training Free* e *BinP* para *Binary Training Periodic*, indicando, respectivamente, a versão para treinamento de sensores livres e periódicos. Todos os resultados apresentados são médias com intervalo de confiança a 99%. O intervalo é representado por barras erros, as quais são por vezes, imperceptíveis.

A Figura 4(a) mostra que o protocolo *Binary Training*, para ambos tipos de sensores, apresenta problemas de convergência. Para tal protocolo, o Índice de Convergência é próximo de 40% para $d_n = 4$. À medida que d_n aumenta, o Índice de Convergência decai exponencialmente, tendendo a valores próximos de 5%. Devido a perda de *beacons control*, o funcionamento do algoritmo *Binary Training* é prejudicado pela atualização indevida dos registradores, sobretudo com o aumento de d_n. Já o protocolo *Flat+*, apresenta um Índice de Convergência em torno de $99,98\%$. A Figura 4(b) mostra a relação entre o Índice de Convergência e o posicionamento real do sensor. Observa-se que os problemas de convergência do *Flat+* são concentrados nas coroas mais externas ($C_i \in [550, 574]$) devido ao alto *PER* experimentado pelos sensores. Note que um sensor localizado nas últimas coroas está quase no limite de transmissão do ator, ou seja, ele recebe menos *beacons*. Para o *Binary Training*, o Índice de Convergência cai rapidamente à medida que o sensor é posicionado mais distante do ator. Isso ocorre porque quanto maior a distância entre o sensor e o ator, maior a probabilidade do sensor perder *beacons control*. A Figura 4(c) mostra como o Erro varia em função de d_n. Note que o *Flat+* obtém um erro superior a 50 coroas (cerca de $500\ m$ de raio). Embora o *Binary Training* apresente um erro menor, este resultado compreende apenas sensores cujo treinamento convergiu (vide Figura 4(a)).

No algoritmo *Flat+*, a condição de parada pode ser obtida erroneamente caso algum *beacon* seja perdido. Também são possíveis erros de convergência quando o sensor experimenta altas taxas de perda de *beacons*, fazendo com que a condição de parada nunca seja obtida. Isto ocorre, sobretudo, com sensores localizados nas coroas mais externas,



(a) Convergência (b) Convergência e posição do sensor



(c) Erro médio de estimativa

Figura 4. Convergência e precisão dos Algoritmos.

ou seja, aqueles que recebem menos *beacons*. No caso do *Binary Training*, a perda de pacotes impossibilita determinar precisamente se o *beacon* recebido foi *control* ou *data*, pois o protocolo não considera que *beacon* carrega esta informação. Note que *Binary Training* assume que um *control* sempre será escutado por ser transmitido na potência máxima. Isso causa erros de convergência, por exemplo, quando o sensor deixa de escutar um par de *beacons*. Tal erro, identificado pela condição $min > max$, se não tratado impossibilita a parada do algoritmo.

6. O protocolo *Strong Flat*

O principal problema nos protocolos estudados está nas suposições sobre o recebimento dos *beacons*. Em condições reais, a recepção de um *beacon* não é garantida mesmo para um receptor dentro do alcance de transmissão do ator. Para verificar se o nó recebe realmente ou não um *beacon* b_i com certa garantia, pode ser necessário escutá-lo mais de uma vez. Com base neste princípio, é proposto o protocolo *Strong Flat*.

A ideia é armazenar informações sobre quantas vezes o *beacon* b_i foi recebido e quantas vezes o *beacon* não foi escutado utilizando, respectivamente, os vetores $H_k[\,]$ e $M_k[\,]$. O princípio está em assumir a recepção do *beacon* b_i apenas se $H_k[i] = \beta_{hit}$ (*beacon* audível). Analogamente, o sensor assume que não recebe o *beacon* b_i se $M_k[i] = \beta_{miss}$ (*beacon* inaudível). O algoritmo também armazena dois registradores: min armazena o menor *beacon* audível e max armazena o maior *beacon* inaudível. Logo, o sensor estará treinado na coroa C_i se b_i é audível e b_{i-1} é inaudível. O comportamento do ator é o mesmo descrito para o protocolo *Flat+*.

O detalhamento do protocolo proposto é apresentado no Algoritmo 6.1. Na primeira recepção, é realizada a sincronização e o *backtracking* para atualizar max (linhas 7-14), assim como no algoritmo *Flat+*. A cada recepção do *beacon* b_c o elemento $H_k[c]$ é atualizado (linha 15). Analogamente, o mesmo ocorre com o elemento $M_k[c]$ cada vez que *beacon* b_c não é ouvido (linhas 25-26). Quando é garantido um *beacon* audível, os valores de $H_k[\,]$ e min são atualizados (linhas 16-20). De forma análoga, quando um *beacon* é inaudível, os valores de $M_k[\,]$ e max são atualizados (linhas 27-31). Desta forma, é garantido que $H_k[min] = \beta_{hit}$ e $M_k[max] = \beta_{miss}$. O sensor estará treinado na coroa C_i se $H_k[i] = \beta_{hit}$ e $M_k[i-1] = \beta_{miss}$ (linhas 21-22 e 32-33). Ao fim de cada ciclo, o sensor entra em inatividade (linhas 34-38). O tempo de espera calculado pelo procedimento $Wait()$ é determinado pelo tipo de sensor. Os sensores livres programam sua próxima ativação para escuta do *beacon* $b_{\lceil \frac{min+max}{2} \rceil}$. Já o sensor periódico ignora os ciclos L_n onde todos os *beacons* a serem transmitidos são audíveis ou inaudíveis (*beacons* b_i, com $i \notin [min, max]$).

Quanto maiores β_{miss} e β_{hit}, maior a necessidade de se escutar o *beacon* de interesse. Contudo, isto implica em um maior tempo em escuta pelos sensores. A escolha dos valores de β_{hit} e β_{miss} é um compromisso entre o tempo total em escuta e a precisão

Algoritmo 6.1 Algoritmo *Strong Flat*

```
1:  ouviu ← falso; treinado ← falso
2:  min ← k − 1; max ← 0;
3:  inicializa cada elemento H_k[ ] ← 0 e M_k[ ] ← 0
4:  enquanto ¬treinado faça
5:      para i ← 0 to d_n − 1 do faça
6:          se recebeu b_c então
7:              se ¬ouviu então
8:                  ouviu ← verdade // primeira recepção
9:                  sincroniza e realiza backtracking
10:                 para h ← 0 até c − 1 faça
11:                     se (M_k[h] ≥ β_miss) então
12:                         max ← h
13:                 para h ← 0 até max faça
14:                     M_k[h] = β_miss
15:             H_k[c] ← H_k[c] + 1
16:             se H_k[c] = β_hit então
17:                 para h ← c + 1 até k − 1 faça
18:                     H_k[h] ← β_hit
19:                 se c < min então
20:                     min ← c
21:             se (min = 0) ou (M_k[min − 1] = β_miss) então
22:                 treinado ← verdade; coroa ← C_min
23:         senão
24:             se ouviu então
25:                 c ← t − 1 − t mod k
26:                 M_k[c] ← M_k[c] + 1
27:                 se M_k[c] = β_miss então
28:                     para h ← max até c − 1 faça
29:                         M_k[h] ← β_miss
30:                     se c > max então
31:                         max ← c
32:                 se H_k[max + 1] = β_hit então
33:                     treinado ← verdade; coroa ← C_max+1
34:         se ¬treinado então
35:             se ¬ouviu então
36:                 espera L_n − d_n slots
37:             senão
38:                 espera Wait() slots
```

de localização. De modo semelhante ao protocolo *Flat+*, o treinamento é possível se $d_n \geq mdc(L_n, k)$, com d_n par.

7. Avaliação de Desempenho do Protocolo *Strong Flat*

Esta seção avalia o desempenho do protocolo proposto. Para isso, é considerado o mesmo cenário, intervalo de confiança e parâmetros definidos na Seção 5. Primeiramente, é apresentado um estudo sobre o imapcto dos valores de β_{miss} e β_{hit}. Em seguida, é feito um estudo comparativo entre o protocolo *Strong Flat* e os protocolos *Flat+* e *Binary Training*. Para as avaliações realizadas, foram utilizadas as métricas Índice de Convergência e Erro definidas anteriormente e as seguintes métricas:

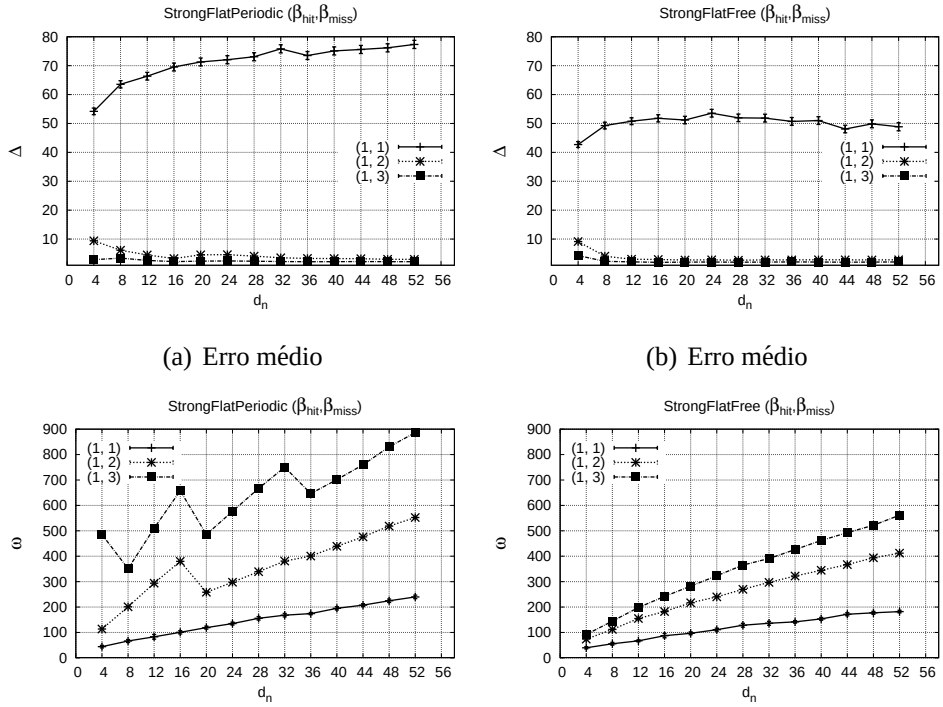
- **Índice de Treinamento (I_n)**: a razão entre o número de sensores treinados com erro máximo de n coroas sobre o número total de sensores;
- **Tempo em Escuta (ω)**: a quantidade de *slots* que o sensor permanece com o rádio em modo de escuta;
- **Tempo Total (τ)**: a quantidade de *slots* decorridos entre a ativação do sensor e o final do treinamento.

7.1. Estudo dos parâmetros β_{hit} e β_{miss}

Esta seção avalia os efeitos dos valores de pares de valores para β_{hit} e β_{miss} no desempenho do protocolo *Strong Flat*. É avaliado o desempenho do protocolo para os pares $(\beta_{hit}, \beta_{miss})$ iguais a $(1, 1)$, $(1, 2)$ e $(1, 3)$. Casos em que $\beta_{hit} > 1$ ocorrem quando um sensor localizado na coroa C_i escuta sinais transmitidos em um nível $j < i$. O estudo deste caso está fora do escopo deste trabalho. É importante observar que o par $(1, 1)$ para sensores periódicos resulta em comportamento similar ao algoritmo *Flat+*. Vale ressaltar que o *Strong Flat Periodic* treina sensores periódicos enquanto o *Strong Flat Free* treina sensores livres.Também é importante ressaltar que em todas as simulações realizadas, o algoritmo executado pelo protocolo *Strong Flat* convergiu.

As Figuras 5(a) e 5(b) enfatizam que os pares $(1, 2)$ e $(1, 3)$ permitem um erro de localização significativamente menor em relação ao obtido para o caso $(1, 1)$. Assim, testar apenas duas vezes se um *beacon* de interesse é inaudível já permite obter melhorias significativas no erro de localização em comparação a um teste único. Note que é observada pouca melhoria no erro de localização ao se passar β_{hit} de 2 para 3. As Figuras 5(c) e 5(d) mostram o impacto do valor de β_{miss} no tempo em escuta. Nesse caso, à medida que o valor de β_{miss} aumenta, o tempo em escuta aumenta. Contudo, o tempo em escuta total é proporcional ao tempo em escuta por ciclo d_n e à quantidade de iterações necessárias para a convergência do algoritmo. Por outro lado, a quantidade de tais iterações decresce com o aumento de d_n já que o tempo em escuta aumenta. Isso justifica o comportamento dente de serra nas curvas para os pares $(1, 2)$ e $(1, 3)$. Observa-se também que os sensores livres apresentam um menor tempo em escuta. Isso ocorre devido a capacidade eles que possuem de controlar dinamicamente o ciclo de trabalho.

A Figura 6 mostra como índice de treinamento de sensores é afetado por β_{miss}. Note que em relação ao caso mais simples de escuta $(1, 1)$, o aumento de β_{miss} permite uma melhoria significativa na quantidade de sensores treinados corretamente na coroa em que estão fisicamente posicionados. A precisão desejada no índice de treinamento depende da aplicação. Algumas aplicações podem tolerar um maior erro. Os resultados

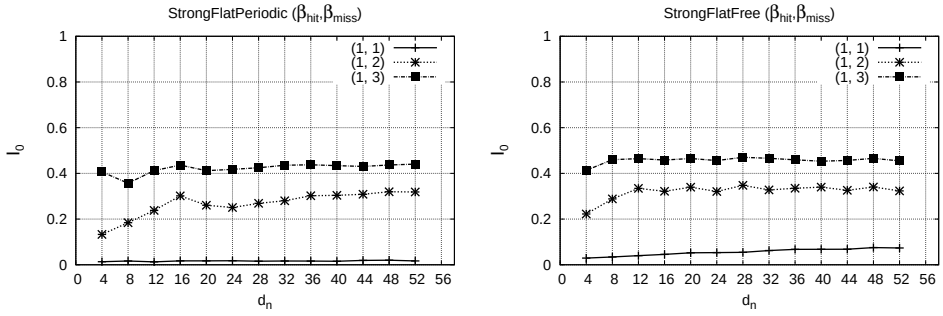


(a) Erro médio (b) Erro médio

(c) Tempo em escuta (d) Tempo em escuta

Figura 5. Efeito dos valores de β_{miss}.

apresentados nesta seção mostram que o par $(1, 2)$ leva a um custo-benefício interessante em relação ao tempo de escuta e o erro de localização para ambos os tipos de sensores.



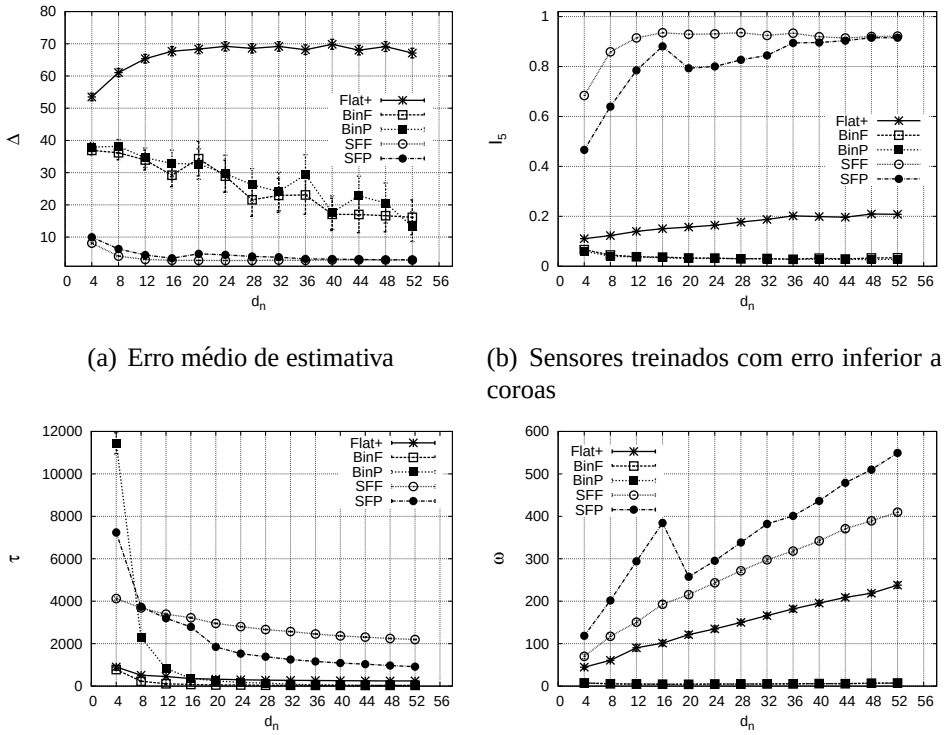
(a) Sensores treinados corretamente (b) Sensores treinados corretamente

Figura 6. Efeito dos valores de β_{miss} sobre o índice de treinamento.

7.2. Comparação entre os Protocolos Estudados

Esta seção compara o desempenho dos protocolos estudados neste artigo. Para os protocolos *Strong Flat* são utilizados $\beta_{hit} = 1$ e $\beta_{miss} = 2$. Para simplificar a apresentação, foi adotada a notação *SFF* para *Strong Flat Free* e *SFP* para *Strong Flat Periodic*.

A Figuras 7(a) e 7(b) mostram, respectivamente, o erro médio de estimativa e o índice de sensores treinados com erro inferior a 5 coroas. O protocolo proposto consegue treinar mais de 80% dos sensores com tal precisão para d_n a partir de 8 *slots*. Além disso,



(a) Erro médio de estimativa

(b) Sensores treinados com erro inferior a 5 coroas

(c) Tempo médio gasto

(d) Tempo médio em escuta

Figura 7. Desempenho dos protocolos

o protocolo proposto permite obter um erro de localização sempre menor do que o obtido com as demais propostas. As Figuras 7(c) e 7(d) mostram, respectivamente, o tempo gasto para o treinamento e o tempo em escuta. Observa-se que o tempo total de treinamento gasto com o protocolo proposto é maior do que o observado para os demais protocolos na maioria dos casos. Mais uma vez, esse resultado é esperado dado que o *Strong Flat* testa mais de uma vez se um *beacon* de interesse é audível.

8. Conclusão

Este trabalho estudou o desempenho dos protocolos assíncronos de treinamento *Flat+* e *Binary Training* para redes de sensores heterogêneas centradas em ator. O canal de comunicação foi modelado para gerar imperfeições na transmissão de *beacons* por parte do ator. Os resultados demonstraram que essas imperfeições levam tais protocolos a problemas de convergência de treinamento e produzem a um aumento nos erros de localização. Para lidar melhor com imperfeições no canal de comunicação, foi proposto o protocolo *Strong Flat*. As avaliações de desempenho do protocolo proposto mostraram que o mesmo não apresenta problemas de convergência e melhora a acurácia de localização em relação aos protocolos estudados.

Referências

Barsi, F., Bertossi, A. A., Betti Sorbelli, F., Ciotti, R., Olariu, S., and Pinotti, M. C. (2009). Asynchronous Corona Training Protocols in Wireless Sensor and Actor Networks. *IEEE Transactions on Parallel and Distributed Systems*, 20(8):1216–1230.

Barsi, F., Bertossi, A. A., Lavault, C., Navarra, A., Olariu, S., Pinotti, M. C., and Ravelomanana, V. (2011). Efficient Location Training Protocols for Heterogeneous Sensor and Actor Networks. *IEEE Transactions on Mobile Computing*, 10(3):377–391.

Bulusu, N., Heidemann, J., and Estrin, D. (2000). Gps-less Low Cost Outdoor Localization for Very Small Devices. *IEEE Personal Communications Magazine*, 7(5):28–34.

Cassano, E., Florio, F., De Rango, F., and Marano, S. (2009). A Performance Comparison Between ROC-RSSI and Trilateration Localization Techniques for WPAN Sensor Networks in a Real Outdoor Testbed. In *Wireless Telecommunications Symposium*, WTS, pages 1–8.

He, T., Huang, C., Blum, B. M., Stankovic, J. A., and Abdelzaher, T. F. (2005). Range-free Localization and its Impact on Large Scale Sensor Networks. *ACM Transactions Embedded Computing Systems*, 4(4):877–906.

IEEE Standard (2001). *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*. IEEE Standard.

Laselva, D., Zhao, X., Meinila, J., Jamsa, T., Nuutinen, J., Kyosti, P., and Hentila, L. (2005). Empirical Models and Parameters for Rural and Indoor Wideband Radio Channels at 2.45 and 5.25 GHz. In *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 1 of *PIMRC*, pages 654–658. IEEE.

Lin, T.-H., Ng, I.-H., Lau, S.-Y., and Huang, P. (2009). Impact of Beacon Packet Losses to RSSI-Signature-Based Indoor Localization Sensor Networks. In *Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 389 –390.

Mao, G., Fidan, B., and Anderson, B. D. O. (2007). Wireless Sensor Network Localization Techniques. *Computer Networks*, 51(10):2529–2553.

Nagpal, R., Shrobe, H., and Bachrach, J. (2003). Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network. In *Proceedings of the 2nd International Conference on Information Processing in Sensor Networks*, IPSN, pages 333–348, Berlin, Heidelberg. Springer-Verlag.

Patwari, N. and Kasera, S. K. (2011). Temporal Link Signature Measurements for Location Distinction. *IEEE Transactions on Mobile Computing*, 10(3):449–462.

Wang, W. and Zhu, Q. (2009). Sequential Monte Carlo Localization in Mobile Sensor Networks. *Wireless Networks*, 15(4):481–495.

Wang, X. and Berger, T. (2008). Spatial Channel Reuse in Wireless Sensor Networks. *Wireless Networks*, 14(2):133–146.

Wang, Z., Tameh., E. K., and Nix, A. R. (2004). Statistical Peer-to-Peer Channel Models for Outdoor Urban Environments at 2 GHz and 5 GHz. In *IEEE 60th Vehicular Technology Conference*, volume 7, pages 5101–5105. IEEE.

Yick, J., Mukherjee, B., and Ghosal, D. (2008). Wireless Sensor Network Survey. *Computer Networks*, 52(12):2292–2330.