Recuperação de Dados por Interpolação para Algoritmos On-Line de Descoberta de Padrões *Flock*

Alternative Title: Data Recover by Path Interpolation for Online Flock Pattern Discovery Algorithms

Vitor Hugo Bezerra
vitorbezerra@uel.br

Denis Evangelista Sanches
sanches.e.denis@gmail.com

Daniel S. Kaster
dskaster@uel.br

Universidade Estadual de Londrina (UEL)
Departamento de Computação (DC)
Londrina-PR, Brasil

RESUMO

A partir da análise de dados espaço-temporais pode-se identificar padrões de movimentação de grupos de objetos, como o padrão *flock*. Este padrão pode ser definido como um número mínimo de entidades dentro de um espaço delimitado por um disco de diâmetro definido que se deslocam juntos por um certo intervalo de tempo. No entanto, enquanto as trajetórias dos diferentes objetos são coletadas, elas podem apresentar irregularidades por problemas, como falha de sistema, falha por passagem em túneis, etc., gerando perdas nas trajetórias coletadas. Uma solução para este problema é a interpolação de pontos, técnica que geometricamente gera pontos correspondentes a pontos faltantes a partir de dados já coletados. Nesse sentido, o objetivo deste trabalho é incluir técnicas de interpolação em algoritmos on-line para o padrão *flock* para o tratamento de *streams* de dados espaço-temporais com perdas com tamanho configurável de memória temporária. A abordagem proposta permite utilizar diferentes métodos de interpolação com baixo *overhead* e bons resultados em termos de precisão. Comparando os resultados utilizando as *streams* originais e as *streams* interpoladas, os experimentos mostraram bons resultados na busca por padrões *flock*, atingindo até 80% de recuperação de respostas perdidas, sem impactar significativamente no custo de execução dos algoritmos.

Palavras-Chave

Bancos de dados espaço-temporais; padrão flock; detecção de padrões; interpolação de trajetória.

ABSTRACT

From the analysis of spatio-temporal data one can identify group patterns of moving objects, for example the flock pattern. This pattern can be defined as a minimal number of entities within a defined disk diameter moving together during a certain time-window. However, as the trajectories of different objects are collected, they may be irregular due to problems such as system failures, passing through tunnels or underground, etc., causing gaps in the collected trajectories. One technique to address this problem is path interpolation, which geometrically generates points corresponding to missing spatio-temporal points based on collected data. In this sense, the objective of this work is to include interpolation techniques in online flock pattern algorithms for the treatment of lossy spatiotemporal data streams using configurable size of temporary memory. Our approach allows employing different interpolation methods with low overhead and good precision results. Comparing results using the original databases and interpolated streams, the experiments showed good results in the search for flock patterns reaching up to 80% recovery of lost answers, without significantly impacting the algorithm execution cost.

CCS Concepts

•**Information systems** → **Data streaming;** Geographic information systems; •**Mathematics of computing** → *Interpolation;*

Keywords

Spatiotemporal databases; flock pattern; pattern detection; path interpolation.

SBSI 2017 June 5^{th} – 8^{th}, 2017, Lavras, Minas Gerais, Brazil

1. INTRODUÇÃO

Tecnologias de geolocalização se tornaram cada vez mais acessíveis devido a sua disseminação em veículos, celulares e demais dispositivos móveis. Essas tecnologias são capazes de monitorar a evolução do posicionamento do objeto móvel através do armazenamento sequencial de suas localizações, pontos, basicamente no formato $p(x_i, y_i, t_i)$, sendo x_i e y_i coordenadas geográficas em t_i, o momento da coleta desse ponto. Em consequência, com maior quantidade de dados espaço-temporais sendo obtida por esses dispositivos, a necessidade de algoritmos eficientes capazes de interpretar essa

grande massa de dados é evidente. A análise desse tipo de dado complexo, embora seja uma operação cara computacionalmente, é de grande importância para a descoberta de comportamentos similares entre objetos como, por exemplo, a detecção de congestionamentos em vias, a migração de animais e o comportamento de pessoas em áreas de grande movimentação [13].

Na literatura são encontradas várias formas de análise de dados espaço-temporais, dentre elas estão as de identificação de padrões de grupos de objetos móveis, especialmente aquelas com representação em disco, que exploram a distância máxima entre um número mínimo de entidades quaisquer não excedendo um diâmetro de disco definido em uma busca. Entre esses padrões da categoria de discos um dos mais representativos é o padrão *Flock*, que é composto por conjuntos com quantidade mínima de objetos que estão espacialmente próximos por um intervalo de tempo pré-definido. Dados um número mínimo de entidades μ, um diâmetro ϵ e um número de instantes de tempo δ, um *flock* é um conjunto de μ ou mais entidades que permanecem por pelo menos δ instantes de tempo consecutivos respeitando o espaço delimitado por um disco de diâmetro ϵ em cada instante de tempo [18].

Contudo, tanto o recebimento quanto a própria coleta desses dados de trajetórias são suscetíveis a falhas e ruídos, podendo ocasionar atrasos e até mesmo a perda de certos pontos de localização, deixando de respeitar a taxa de amostragem dos pontos do respectivo dispositivo. Quando essas incertezas, ou *gaps* (lacunas), são acidentais, causadas por problemas como falhas em sistemas e dispositivos, por passagens por túneis ou subterrâneos, etc., há um *hole* (buraco) na trajetória monitorada [11]. Como resultado, por exemplo, os algoritmos na literatura para identificação dos padrões *flock* sofrem sensível impacto devido à restrição sequencial temporal dos objetos para participarem de possíveis *flocks*. Na prática, esses *holes* podem resultar na não identificação de um grupo de animais migrando, um grupo de bandidos praticando assaltos em uma região ou um grupo de turistas se movendo, por exemplo.

Tanto para o tratamento desses *holes* quanto para a diferença da taxa de coleta dos pontos de localização, implementações exigem considerações a respeito dessas incertezas e aplicação de técnicas de interpolação adequadas [12]. A interpolação de pontos consiste em, a partir de determinados pontos coletados, interpolar e inserir pontos nas lacunas de trajetórias. Diversos trabalhos na literatura [19, 5, 21, 10, 17] utilizam a interpolação linear, por exemplo, para criar aproximações dessas localizações faltantes, a fim de garantir o sincronismo na taxa de amostragem. No entanto, não há trabalhos focados na análise do impacto desses *gaps* em trajetórias e nem na avaliação da utilização de técnicas de interpolação para identificação de *flocks*.

Neste contexto, o objetivo deste trabalho é propor um método capaz de tratar o problema de detecção de *flocks* em trajetórias sujeitas a dados com variação de taxa de amostragem e/ou *holes*. O método proposto consiste em inserir técnicas de interpolação de pontos em algoritmos para a detecção on-line do padrão *flock*, ou seja, possibilitando o tratamento de *streams* de dados de trajetórias de objetos móveis. Nossa proposta armazena dados recebidos em instantes subsequentes de tempo (janela), guardando-os em estruturas de dados (*buffers*), e realizando a interpolação dos pontos faltantes identificados nesses *buffers*.

Experimentos realizados em trajetórias com pontos de localização retirados mostram que a proposta obteve uma grande recuperação de resultados perdidos quando comparados com os resultados com as trajetórias originais completas. Neste trabalho são reportados resultados que confirmam que a taxa de respostas perdidas foi reduzida em até 80%, embora o uso de interpolação seja sujeito à inclusão de falso-positivos que, como analisado neste trabalho, são respostas satisfatórias em boa parte dos casos, visto que são similares às não encontradas.

O restante deste artigo está organizado da seguinte maneira. A Seção 2 apresenta os conceitos básicos relacionados a este trabalho. A Seção 3 apresenta uma análise de sensibilidade dos algoritmos avaliados frente à falta de dados e a proposta de inclusão da técnica de interpolação. Na Seção 4 são apresentados os resultados obtidos e, por fim, na Seção 5, a conclusão e considerações finais.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 Padrão *Flock*

Na literatura são encontradas diversas pesquisas a respeito de descoberta de padrões em dados de objetos móveis. Destas, muitas focam na identificação de padrões de grupos desses objetos, como no estudo da movimentação de animais, com a utilização de coleiras especiais com GPS, ou na análise do tráfego de veículos, tendo suas posições de localização coletadas a cada intervalo de tempo. Essas pesquisas são importantes, pois destes padrões de grupos de objetos podem ser identificados, no caso dos animais, comportamentos de movimentos migratórios e até mesmo de extinção [2] [15].

Um desses padrões, foco deste trabalho, é o padrão *flock* (*flock pattern*), possuindo definições variadas na literatura. Um exemplo é a definição usada no trabalho [7], em que se considera somente um instante de tempo. No entanto, a mais abrangente e mais utilizada na literatura define um *flock* como um grupo de pelo menos μ entidades movendo-se respeitando uma área máxima de um disco de diâmetro ϵ por um intervalo δ de tempo [18, 3, 1, 6, 14, 15]. Seguindo esta linha, a formalização do padrão *flock* utilizada neste trabalho é dada pela definição proposta por [18], reproduzida na Definição 1.

Definição 1 (Padrão *Flock*). *Dado um conjunto de trajetórias $\mathcal{T}$, um número mínimo de trajetórias $\mu > 1$ ($\mu \in \mathbb{N}$), uma distância máxima $\epsilon > 0$ definida por uma métrica de distância d e uma duração mínima de $\delta > 1$ instantes de tempo ($\delta \in \mathbb{N}$), um padrão **Flock**(μ,ϵ,δ) reporta um conjunto $\mathcal{F}$ contendo todos os flocks f_k, que são conjuntos de trajetórias de tamanho maximal tais que: para cada $f_k \in \mathcal{F}$, o número de trajetórias é maior ou igual a μ e existem δ instâncias de tempo consecutivas $t_j, \ldots, t_{j+\delta-1}$ em que existe um disco com centro $c_k^{t_i}$, de diâmetro ϵ, cobrindo todos os pontos das trajetórias de $f_k^{t_i}$, que é flock f_k no instante t_i, $j \leq i \leq j + \delta$.*

Neste trabalho, para a identificação do padrão *flock*, optou-se pela utilização do algoritmo BFE. O algoritmo BFE (***B**asic **F**lock **E**valuation*), proposto em [18], utiliza um índice baseado em grade para processar os dados das trajetórias em cada instante de tempo, sendo o primeiro proposto on-line para descoberta de padrões *flock* em tempo polinomial.

2.2 Técnicas de Interpolação

Dados de trajetórias de objetos móveis podem ser incertos e incompletos, ou seja, imprecisos, em decorrência de vários motivos. Exemplos vão desde imprecisões na coleta dos pontos da trajetória, pois os dispositivos utilizados podem ser imprecisos, até a dificuldade de captura de dados de um objeto que está se movendo constantemente. Já que sua posição só pode ser guardada em um certo período de tempo essa informação é perdida caso não haja a sincronização no tempo limite permitido [22, 11].

Os motivos podem se estender também a problemas de comunicação via rede, causados por falta de sinal de antena, falha em equipamentos ou até mesmo falha humana ao desligar o aparelho de coleta em algum momento, por exemplo. A incerteza também pode advir da natureza do conjunto de dados. Algumas aplicações, por exemplo, não estão interessadas na trajetória inteira, mas sim em pequenas partes, referentes a intervalos de tempo maiores, específicos. No entanto, essa taxa de amostragem de pontos de localização pode ser aquém da necessária para identificação de padrões como o *flock* [11].

Na literatura há várias técnicas para o tratamento desses dados, incluindo as baseadas em busca por abrangência (*range queries*) para minimizar o impacto das incertezas das localizações [22]; o aumento no tempo entre um ponto e outro retirando alguns desnecessários (limpeza da trajetória, focando em sincronização temporal dos pontos); a utilização de filtros como o *Kalman Filter* [8] e outras. As técnicas abordadas neste trabalho para tratamento de incertezas realizam interpolação de pontos. Interpolação é a ação de estimar um novo ponto em um instante de tempo não contido na amostra de dados – *gap* na trajetória – com base nos dados existentes na amostra [9]. Para a sua utilização são necessários pontos de um objeto, anteriores e/ou posteriores ao instante de tempo desejado, i.e., faltante, estimando o posicionamento do objeto móvel no instante por meio de equações matemáticas.

A escolha dos métodos concentrou-se em abordagens aplicáveis a dados "brutos" de trajetória, isto é, sem considerar informação adicional, como malha viária e demais informações semânticas. Os escolhidos foram os seguintes.

- **Interpolação Linear** – A interpolação linear calcula a localização estimada de um objeto por uma linha reta entre dois pontos reais coletados. Conhecido por ser um método simples de ser implementado, possui resultados muito bons para faltas sensíveis de dados, além de ser utilizado recorrentemente na literatura como parâmetro de comparação em estudos.

- **Interpolação por *Random Walk* Restrito (RW)** – A interpolação por *Random Walk* restrito (*Constrained Random Walk*) consiste em realizar uma interpolação independente da posição anterior do objeto móvel na trajetória. A interpolação é criada a partir de amostras aleatórias de duas distribuições: a distribuição do comprimento de passo (l) e a distribuição do ângulo de rotação (θ). RW é recomendado em trajetórias em que o objeto observado tem um caminho difícil de se prever, como por exemplo, animais que não têm um trajeto usual, pois não seguem um caminho direto até uma localização, como, por exemplo, macacos [20].

- **Interpolação pela Curva de Bézier** – A interpolação com curva cúbica de Bézier requer a definição de quatro pontos âncora. Método bastante recomendado para trajetórias cheias de curvas, e.g., animais marítimos [16].

3. INCLUSÃO DE INTERPOLAÇÃO EM ALGORITMOS DE DETECÇÃO DE *FLOCKS*

Incertezas nos dados de entrada impactam significativamente os algoritmos de detecção de *flocks*, destacando-se a perda de pontos de localização coletados, principalmente em algoritmos on-line. Esse *gap* na trajetória sendo processada em determinado instante de tempo acarreta na eliminação desta em qualquer possível candidato a *flock* durante a janela temporal sendo processada, visto a restrição de consecutividade temporal dada na definição do padrão *Flock* (Definição 1). Como impacto, os algoritmos podem identificar *flocks* com menos objetos que o na realidade, devido ao descarte destes com *gaps* em suas trajetórias, ou até mesmo deixar de reportar outros *flocks*, caso a remoção desses objetos descarte possíveis candidatos por terem quantidade insuficiente de objetos para formar o padrão.

Qualquer algoritmo de identificação de padrões *flock* sem o devido tratamento é sensível a esse erro, em especial os on-line, incluindo o BFE, utilizado neste trabalho, já que são limitados a trabalhar apenas com *stream* de dados. Esta seção apresenta uma análise do comportamento do algoritmo on-line BFE quando as trajetórias dos objetos em estudo possuem *holes*, retirando-se aleatoriamente porcentuais pré-definidos de pontos. Em seguida, é apresentada a proposta de inclusão de técnicas de interpolação nesses algoritmos.

3.1 Análise da Sensibilidade dos Algoritmos do Padrão *Flock* à Falta de Dados

Conforme abordado anteriormente, é sabido que a falta de pontos em certos intervalos de tempo pode impactar significativamente nos resultados obtidos pelos algoritmos de detecção do padrão *flock*. Contudo, nenhum trabalho havia analisado quantitativamente o impacto da falta de pontos nessas respostas. Desta forma, a seguir é apresentada uma análise a respeito desse aspecto, confirmando que de fato os algoritmos são muito sensíveis à falta de dados.

Para simular a realidade das incertezas das trajetórias em relação à perda de dados de localização e analisar seu impacto, optou-se por fazer uma retirada controlada de pontos em três conjuntos de dados conhecidos. Desta forma, as respostas identificadas nos conjuntos de dados originais formam a "regra ouro", e os conjuntos com pontos retirados simulam situações de falta de dados, possibilitando avaliar a sensibilidade dos algoritmos quanto a esse cenário. O método de retirada controlada de pontos foi desenvolvido em Python[1] e trata separadamente as trajetórias de cada objeto do conjunto de dados. Para cada trajetória é retirado o mesmo percentual de pontos, fornecido como parâmetro, garantindo a homogeneidade da retirada dos pontos para todos os objetos do conjunto de dados, tratando a falta de pontos como um comportamento inerente ao processo de captura do posicionamento dos objetos, e não específico de determinados objetos.

Para comparação das respostas obtidas com as diferentes entradas de dados, o método desenvolvido inicialmente armazena a regra ouro de cada conjunto de dados original

[1] www.python.org

em arquivos diferentes, de forma que em cada linha do respectivo arquivo contenha o intervalo de tempo de cada *flock* identificado, seguido dos objetos contidos no mesmo, sendo o arquivo ordenado pelo tempo de início desse intervalo. Para as entradas de dados com a simulação de perda de pontos, foram guardadas as respostas obtidas de forma análoga ao feito nos conjuntos de dados originais. Na sequência, para cada configuração $Flock(\mu, \epsilon, \delta)$ de parâmetros e para cada percentual de dados perdidos, foi feita a comparação resultado a resultado, com *flocks* que possuem o mesmo início, com sua respectiva regra ouro, onde:

(a) **Falso-Negativos (FNs):** são os *flocks* que deveriam ter sido reportados, mas não foram;

(b) **Falso-Positivos (FPs):** são os *flocks* reportados que divergem da regra ouro.

A análise utilizou os conjuntos de dados originais dos trabalhos [18, 14, 15] que propuseram os algoritmos BFE e PSI, bem como suas variações, *Buses*, *Caribous* e *Cars*. *Buses* contém 66.096 posições de ônibus se movendo na cidade de Atenas, Grécia. O segundo conjunto de dados é denominado *Caribous* e foi gerado a partir da movimentação de 43 cervídeos na região noroeste do Canadá, com um total de 15.796 localizações. Já o conjunto *Cars*[2] é composto de 134.264 posições coletadas a partir da movimentação de 183 carros privados movimentando-se em Copenhague, Dinamarca.

O processo da análise inicial foi executado da seguinte forma. De cada conjunto de dados foram derivados três outros conjuntos, considerando as porcentagens de retirada de pontos 1%, 5% e 10%. Para melhor avaliar o impacto da retirada de pontos das trajetórias nos resultados, foram executadas 30 repetições de cada conjunto de dados com cada taxa de retirada de pontos (1%, 5% e 10%), sempre retirando-se pontos de forma aleatória em cada repetição.

A Figura 1 mostra que os algoritmos são muito sensíveis à falta de pontos, pois mesmo a falta de 1% dos pontos faz com que os algoritmos deixassem de reportar entre 20% e 60% das respostas esperadas. A retirada de 5% e 10% dos pontos de cada trajetória faz com que praticamente todas as respostas dos algoritmos sejam perdidas. Percebe-se, ainda, que com o crescimento das variáveis μ e δ, há um aumento no número de respostas que são perdidas, pois amplia-se a probabilidade de um ponto faltante dissolver um *flock* inteiro em decorrência da natural maior seletividade das respostas com o aumento desses parâmetros. Já a variação do parâmetro ϵ mostrou-se menos sensível à retirada dos pontos de localização.

Os experimentos de validação da proposta deste trabalho contemplam retiradas de 10%, 20% e 30%, simulando uma grande perda de dados mas que ainda seria possível alcançar resultados satisfatórios com a interpolação desses pontos faltantes utilizando o método proposto a seguir.

3.2 Proposta de Inclusão de Interpolação nos Algoritmos On-Line de *Flocks*

Como observado na seção anterior, a lacuna do posicionamento de um objeto móvel em um único instante de tempo pode impedir sua inclusão em um *flock*. Isto porque os algoritmos de detecção do padrão *flock* on-line recebem os dados da posição dos objetos a cada instante de tempo. Para contornar essa limitação, a abordagem adotada neste trabalho foi utilizar algoritmos de interpolação de pontos para tornar os algoritmos mais robustos com relação a esse aspecto. Para isso, técnicas de interpolação foram implementadas dentro do algoritmo BFE, pois os algoritmos disponíveis de interpolação de uma forma geral consideram que todo o conjunto de dados está disponível, premissa que não é válida para abordagens on-line.

O método proposto armazena as posições dos objetos em dados instantes de tempo em estruturas de dados, chamadas aqui de *buffers*. Cada *buffer* guarda as posições de todos os objetos em um certo tempo t, formando uma espécie de janela de tempo da movimentação das trajetórias. A janela é centralizada no *buffer atual*, que é o instante de tempo a ser verificado para a necessidade de interpolar pontos faltantes. Os demais *buffers* são utilizados para:

(a) **identificar a perda de uma posição**: neste aspecto, assume-se que se uma trajetória contém pontos antes e depois do instante de tempo atual, então há um *hole* no instante atual a ser interpolado; e

(b) **gerar um ponto interpolado**: os pontos nos instantes precedente(s) e subsequente(s) são utilizados para estimar o posicionamento do objeto no instante de tempo em que ocorreu a falha.

Ao processar o *buffer atual* verifica-se se há pontos que estejam no *buffer* $t-1$ e em algum outro *buffer* futuro, mas não o atual, caracterizando um *hole* a ser interpolado. Detectado esse *gap* na trajetória do objeto, então esse ponto de localização é interpolado com base nos dados desse objeto nos *buffers* de tempos anteriores e futuros. Após a análise deste *buffer*, ele é liberado e os dados seguem para o algoritmo de detecção de *flock*, no caso o BFE.

Uma ilustração do método pode ser vista na Figura 2, em que a proposta é ilustrada com a utilização de uma janela de cinco *buffers*, representados na figura pelos cilindros. Na imagem, o *buffer* atual é o que está sendo analisado e contém as posições dos objetos em um tempo t. As posições neste *buffer* são comparadas com as posições apresentadas no *buffer* passado, que contém as posições do instante de tempo $t-1$. Se houver um objeto que reportou uma posição no tempo anterior e não reportou no tempo t, verifica-se a existência de outro ponto de localização desse objeto no(s) *buffer(s)* futuro(s). Caso exista algum até o fim da janela atual o ponto será interpolado com base nas posições do objeto em $t-1$ e no *buffer(s)* futuro(s) encontrado(s).

O número de pontos de controle para realizar a interpolação depende do método utilizado. Por exemplo, a interpolação linear e a interpolação por *random walk* restrito precisam de um ponto anterior ao ponto a ser interpolado e um ponto posterior. Já a interpolação pela curva de Bézier precisa de dois pontos anteriores e dois posteriores. Foram testados *buffers* contendo o mínimo de dados para serem utilizados pelas interpolações (3 para linear e *random walk* restrito, e 5 para a interpolação curva de Bézier), além de 7 e 9 *buffers*. Vale ressaltar que a abordagem proposta insere um *delay* (atraso) na identificação do padrão *flock*, que depende do número de *buffers* posteriores ao instante atual. No entanto, conforme resultados deste trabalho, processar com uma janela de 9 *buffers* insere um *delay* quase irrelevante ao tempo total de execução do algoritmo on-line, além de convergir a resultados satisfatórios na recuperação de *flocks* perdidos, decorrentes dos *holes* nas trajetórias.

[2] www.daisy.aau.dk

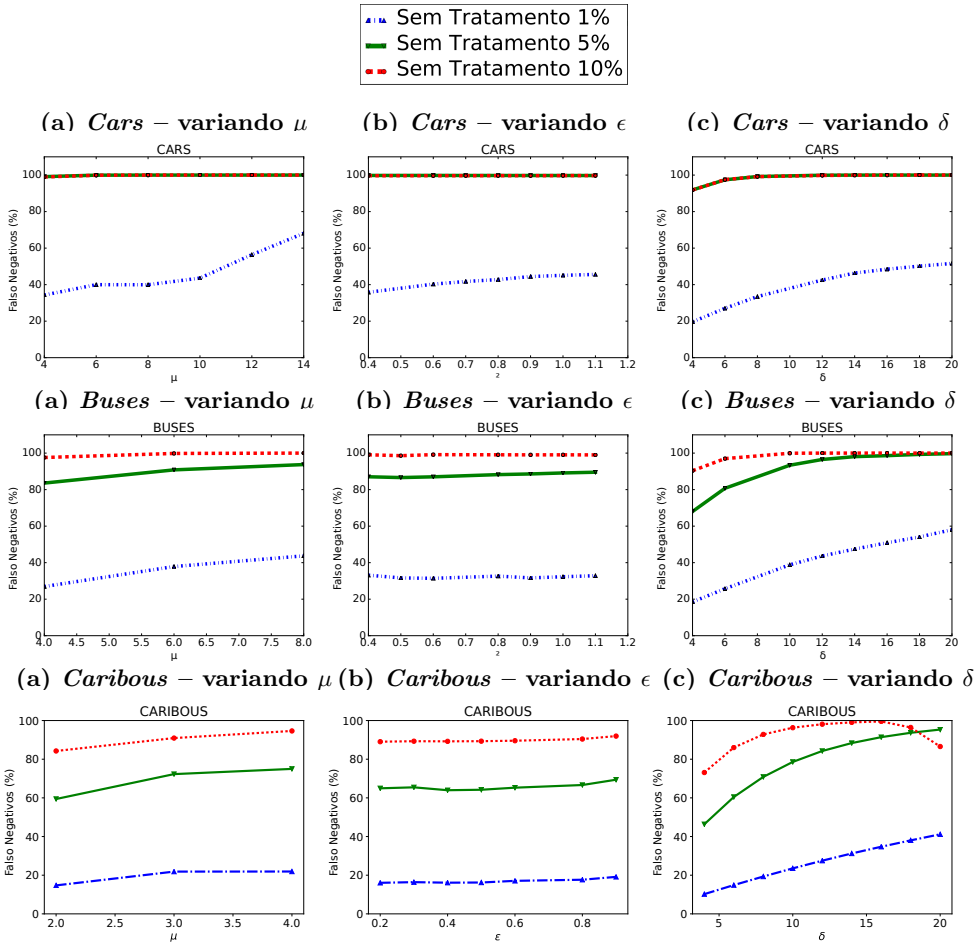

Figure 1: Taxa de FNs gerado pelo algoritmo BFE mediante diferentes taxas de perdas/retirada de pontos. *Cars* (1ª linha), *Buses* (2ª linha) e *Caribous* (3ª linha) variando μ (cardinalidade – 1ª coluna), ϵ (diâmetro dos discos – 2ª coluna) e δ (duração – 3ª coluna).

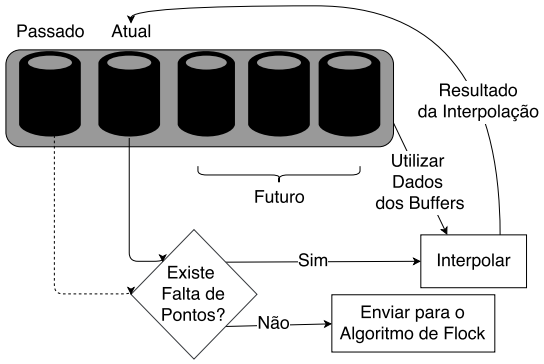

Figure 2: Ilustração do método proposto com a utilização de cinco *buffers*.

4. AVALIAÇÃO EXPERIMENTAL DA PROPOSTA

Esta seção descreve os resultados obtidos sobre os três conjuntos de dados *Cars*, *Buses* e *Caribous*, avaliando a eficácia e eficiência da proposta. Foram realizadas baterias de experimentos seguindo o mesmo método de teste e os mesmos parâmetros utilizados na avaliação de sensibilidade dos algoritmos on-line do padrão *flock* (Seção 3.1). A análise foi guiada por uma série de perguntas de análise elaboradas para avaliar o método proposto.

A primeira pergunta que desejou-se responder na avaliação foi: "*o método proposto reduz a taxa de perdas (falso-negativos)?*" A Figura 3 mostra as taxas de acerto da proposta utilizando 5 *buffers*, considerando 10% de perda no conjunto *Cars*, *Buses* e *Caribous*, para as três técnicas de interpolação implementadas: linear, *random walk* restrito e curva de Bézier. Nota-se que a taxa de falso-negativos caiu consideravelmente em relação à situação em que não foi realizada interpolação (Figura 1) para os três conjuntos (figuras 3(a–c), 3(d–f) e 3(g–i)), com destaque para a interpolação linear. A interpolação linear inclusive possibilitou a redução da taxa de falso-negativos de quase 100% no conjunto *Cars* para pouco menos de 10% e a taxa do conjunto *Buses* de mais de 80% também para menos de 10%. A curva de Bézier teve os piores resultados, parte devido à natureza dos conjuntos de dados considerados, baseados em malha viária, e outra por, necessariamente, exigir que haja pontos de localização nos dois *buffers* futuros para que o ponto

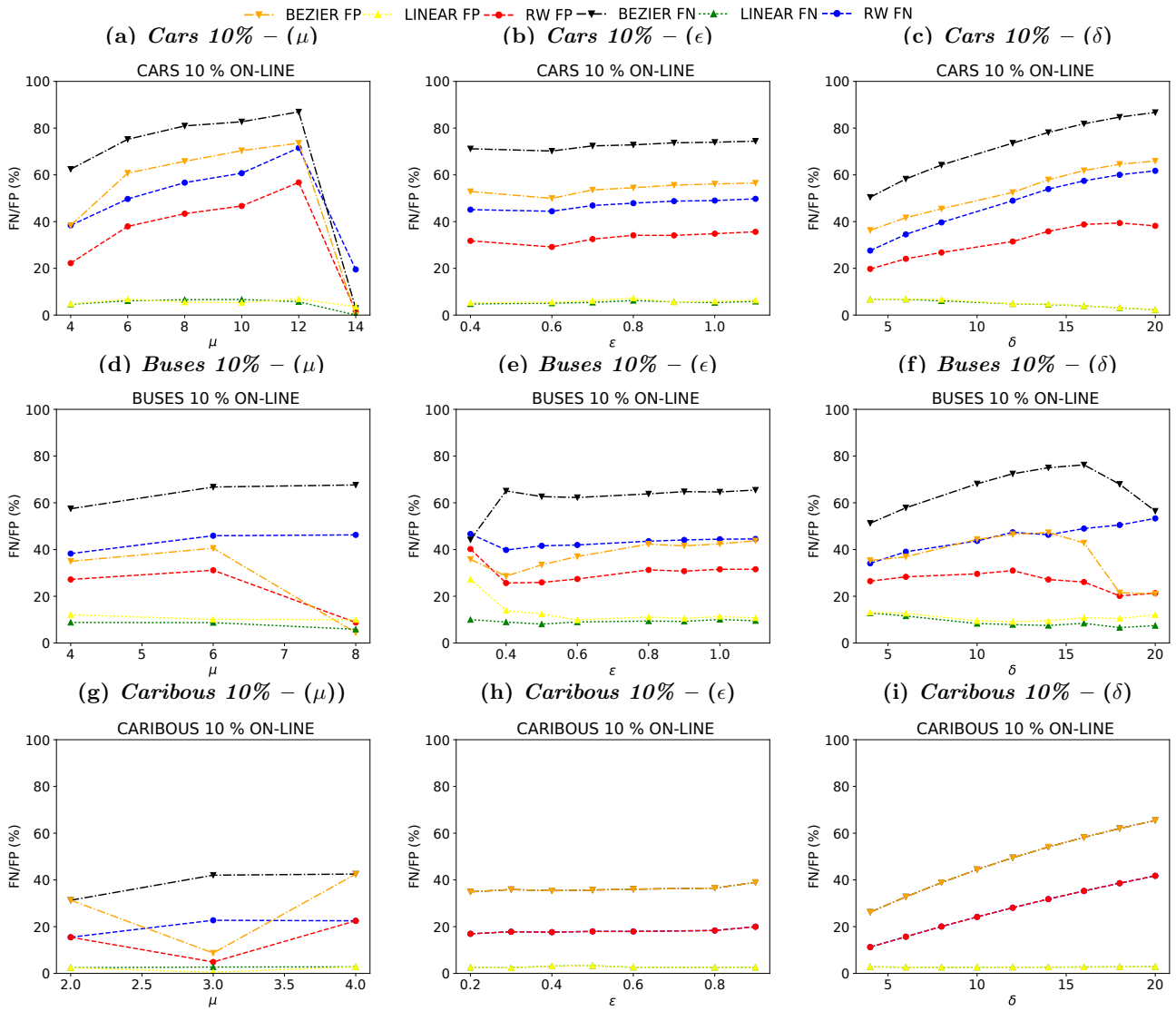

Figure 3: Taxa de acerto dos métodos de interpolação on-line, variando μ (cardinalidade – 1ª coluna), ϵ (diâmetro dos discos – 2ª coluna) e δ (duração – 3ª coluna). *Cars* com 10% de perda – FNs/FPs (1ª linha), *Buses* com 10% de perda – FNs/FPs (2ª linha) e *Caribous* com 10% de perda – FNs/FPs (3ª linha).

faltante, *buffer* atual, seja interpolado, diminuindo assim a chance de interpolação de um *hole* real. Entre a linear e a de Bézier, encontra-se a *random walk* restrito, que não obteve resultados melhores pois sua interpolação interfere muito na localização dos objetos criando *flocks* diferentes ou retirando objetos de respostas existentes. Vale ressaltar que, em experimentos anteriores utilizando janelas com mais *buffers*, a interpolação de pela curva de Bézier apresentou resultados consideravelmente melhores que a RW restrito [4]. *Flocks* com menor tempo de duração e menor quantidade de objetos foram os que tiveram melhores recuperações de respostas, exceto para μ relativamente grandes, visto que até mesmo para os conjuntos de dados originais identificam poucos *flocks*. Já a diferença do tamanho do diâmetro do *flock* não resultou em grandes diferenças.

A segunda pergunta de análise foi: "*quanto os métodos de interpolação introduzem de falso-positivos na detecção de flocks?*" Observa-se que o uso de interpolação é sujeito à geração de falso-positivos, isto é, *flocks* que não existem de fato segundo a regra ouro, mas foram detectados após a inclusão dos pontos de localização interpolados que diferem dos originais. Os gráficos das figuras 3(a–c), 3(d–f) e 3(g–i)) mostram que os falso-positivos fizeram a taxa total de erros dobrar na maioria dos casos. Entretanto, a taxa de erro global ainda é consideravelmente inferior à obtida sem interpolação (Figura 1), sustentando a eficácia da proposta.

A terceira questão foi: "*o número de buffers impacta na qualidade dos resultados?*" A Figura 4 mostra a taxa média de falso-negativos para certos parâmetros (μ, ϵ, δ), variando o tamanho do *buffer* na interpolação linear. Para menores porcentagens de retirada de pontos os resultados são os mesmos, mas a partir da retirada de 10% o tamanho do *buffer* começa a influenciar na recuperação de pontos, pois há *gaps* nas trajetórias, necessitando utilizar pontos mais ao futuro

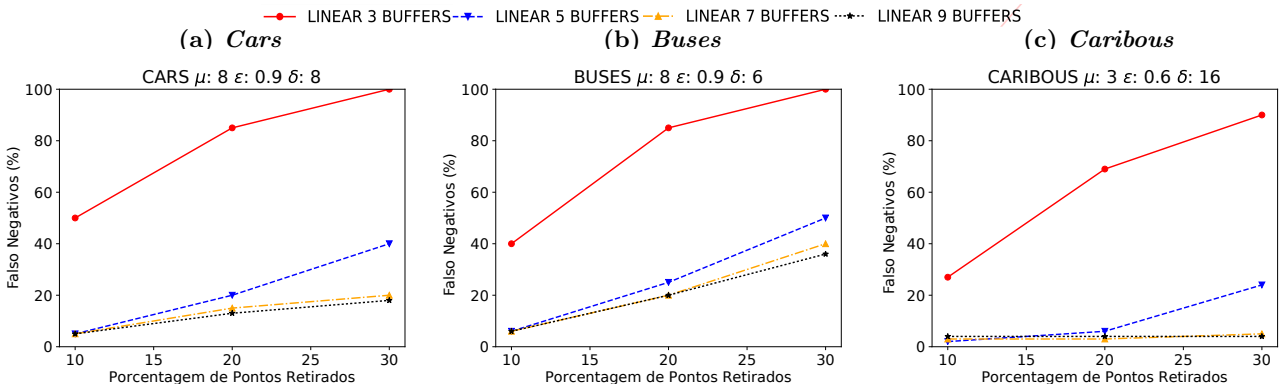

Figure 4: Taxa de acerto do método de interpolação on-line linear, variando retirada de pontos de dados, *Cars* (1ª coluna) FNs, *Buses* (2ª coluna) FNs e *Caribous* (3ª coluna) FNs.

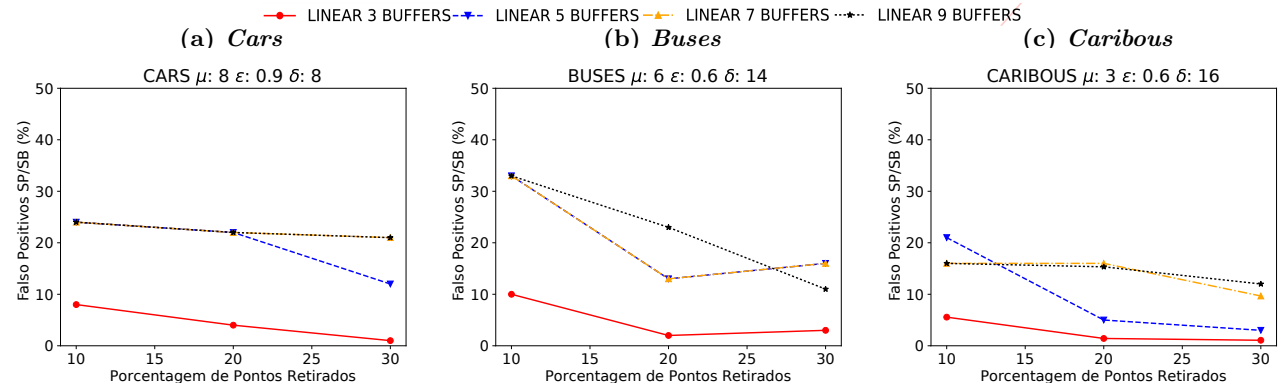

Figure 5: Taxa de FPs superconjunto ou subconjunto de FNs do método interpolação de on-line linear, variando retirada de pontos de dados, *Cars* (1ª coluna), *Buses* (2ª coluna) e *Caribous* (3ª coluna).

para realizar a interpolação, devido ao aumento da probabilidade da retirada de pontos acarretar na inserção de *holes* consecutivos. A utilização de 9 *buffers* obteve os melhores resultados, reduzindo em até 20% a perda de respostas para um conjunto com bem mais lacunas, dos avaliados na Seção 3.1. No entanto, a utilização de 7 *buffers* teve resultados similares à utilização de 9, o que gera um menor *delay* ao usuário e com precisão semelhante. A execução com janela de comprimento 5 obteve resultados medianos batendo a recuperação de resposta ao se comparar com a de apenas 3 *buffers*, entretanto, com considerável diferença entre a utilização de 7 e 9. Por último, utilizando 3 *buffers*, como esperado, obteve o pior resultado pois, com a retirada de 30% e 20% praticamente inviabiliza a chance de interpolações.

A quarta questão foi: "*quão ruins são os falso-positivos retornados para aplicações que utilizam o padrão flock?*" Neste sentido, a Figura 5 apresenta uma análise dos falso-positivos para certos parâmetros (μ, ϵ, δ) nos conjuntos de dados, variando o tamanho do *buffer* na interpolação linear, apenas em virtude da limitação de espaço deste trabalho. Com base em uma função de similaridade aplicada sobre os falso-positivos e os falso-negativos verificou-se que mais de 30% dos falso-positivos possuem ao menos 75% de similaridade em relação os falso-negativos, ou seja, são subconjuntos ou superconjuntos destes. Isso demonstra que dentre os falso-positivos há *flocks* identificados mesmo que parcialmente, demonstrando um maior grau de recuperação de respostas.

Por fim, a última pergunta de análise foi: "*qual o impacto, em termos de tempo de execução, da interpolação de pontos no método proposto?*" Em relação ao custo adicional em tempo de execução da proposta sobre o algoritmo BFE, verificou-se ser praticamente irrelevante, visto o *overhead* mínimo identificado ao se comparar as execuções com as *streams* interpoladas e as com o algoritmo original. Ressalta-se, no entanto, que a interpolação linear é a mais eficiente por ser facilmente calculada, sendo, por exemplo, cerca de 25% mais rápida que a curva de Bézier.

Os resultados apresentados confirmam a eficácia do método proposto para recuperar pontos para aprimorar os resultados de descoberta on-line do padrão flock.

5. CONCLUSÃO

A maior quantidade de dados espaço-temporais tem criado uma demanda de algoritmos de descoberta de padrões. Dentre os vários padrões de movimentação que podem ser encontrados na literatura, o padrão *flock* é um dos que merece destaque. No entanto, na coleta dos pontos de localizações desses objetos móveis, por inúmeros fatores, falhas e ruídos podem ocorrer, como na perda de certas localizações, que, como avaliado neste trabalho, afetam consideravelmente os algoritmos de detecção de padrão *flock*, em especial os on-line. Neste contexto, este trabalho apresentou uma proposta para o tratamento de falta de pontos em trajetórias por meio

de técnicas de interpolação de pontos. O método proposto inclui três tipos de interpolação: linear, *constrained random walk* e Bézier, que foram adaptados em algoritmos de descoberta do padrão *flock* para realizar interpolação on-line, focado na manipulação de *streams* de dados. Nessa estratégia, devido ao fato de poder existir recorrentes *gaps* nas trajetórias, e pelo tamanho da janela de processamento limitado, os pontos previamente interpolados também são utilizados para a criação de novos pontos.

Também foram realizados diversos experimentos para avaliar o impacto da perda de dados de posicionamento e a eficácia e eficiência da proposta. Os resultados mostraram que os algoritmos são muito sensíveis visto que uma pequena perda nos pontos já impacta significativamente. Os resultados obtidos com a aplicação da proposta possibilitaram um aumento expressivo na qualidade dos resultados, em particular com a utilização da interpolação linear. Tudo isso, sem acréscimo significante no tempo de execução do algoritmo on-line, mesmo trabalhando com 9 *buffers*, a maior janela experimentada neste trabalho.

Trabalhos futuros incluem a aplicação de outras técnicas de interpolação e que não dependam de dados futuros, um teste real da proposta e a avaliação de outras variáveis, como o impacto da proposta no consumo de memória.

6. REFERENCES

[1] H. Arimura and T. Takagi. Finding All Maximal Duration Flock Patterns in High-dimensional Trajectories. *Manuscript, DCS, IST, Hokkaido University, Apr*, 2014.

[2] G. Balme and L. Hunter. Mortality in a protected leopard population, Phinda Private Game Reserve, South Africa: a population in decline. *Ecological Journal*, 2004.

[3] M. Benkert, J. Gudmundsson, F. Hübner, and T. Wolle. Reporting flock patterns. volume 41, pages 111–125. Elsevier, 2008.

[4] V. H. Bezerra and D. S. Kaster. Inclusão de técnicas de interpolação de pontos em algoritmos de descoberta on-line do padrão flock. In *Anais da XIII Escola Regional de Banco de Dados*, pages 57–66. SBC, 2017.

[5] T. L. C. da Silva, K. Zeitouni, and J. A. de Macedo. Online Clustering of Trajectory Data Stream. *Proceedings of the 17th IEEE International Conference on Mobile Data Management*, pages 112–121, 2016.

[6] J. Gudmundsson and M. van Kreveld. Computing longest duration flocks in trajectory data. In *Proceedings of the 14th ACM Int. Symposium on Advances in Geographic Information Systems*, page 35, New York, New York, USA, 2006. ACM Press.

[7] J. Gudmundsson, M. van Kreveld, and B. Speckmann. Efficient Detection of Motion Patterns in Spatio-temporal Data Sets. In *Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems*, pages 250–257, 2004.

[8] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on signal processing*, 50(2):425–437, 2002.

[9] L. Li and P. Revesz. A comparison of spatio-temporal interpolation methods. In *Geographic Information Science*, pages 145–160. Springer, 2002.

[10] Z. Li, B. Ding, J. Han, and R. Kays. Swarm: Mining Relaxed Temporal Moving Object Clusters. *Proceedings of the VLDB Endowment*, 3(1-2):723–734, sep 2010.

[11] C. Parent, S. Spaccapietra, C. Renso, G. Andrienko, N. Andrienko, V. Bogorny, M. L. Damiani, A. Gkoulalas-Divanis, J. Macedo, N. Pelekis, Y. Theodoridis, and Z. Yan. Semantic trajectories modeling and analysis. *ACM Computing Surveys*, 45(4):42:1–42:32, Aug. 2013.

[12] D. Pfoser and C. S. Jensen. Capturing the Uncertainty of Moving-Object Representations. In R. H. Güting, D. Papadias, and F. Lochovsky, editors, *Proceedings of 6th International Symposium on Advances in Spatial Databases*, pages 111–131. Springer, 1999.

[13] K. C. V. Rao, A. Govardhan, and K. C. V. Rao. Spatiotemporal Data Mining: Issues, Tasks And Applications. *International Journal of Computer Science & Engineering Survey*, 3(1):39–52, feb 2012.

[14] P. S. Tanaka, M. R. Vieira, and D. S. Kaster. Efficient Algorithms to Discover Flock Patterns in Trajectories. *XVI Brazilian Symposium on Geoinformatics*, 1(XVI):56–67, 2015.

[15] P. S. Tanaka, M. R. Vieira, and D. S. Kaster. An Improved Base Algorithm for Online Discovery of Flock Patterns in Trajectories. *Journal of Information and Data Management*, 7(1), 2016.

[16] Y. Tremblay, S. A. Shaffer, S. L. Fowler, C. E. Kuhn, B. I. McDonald, M. J. Weise, C.-A. Bost, H. Weimerskirch, D. E. Crocker, M. E. Goebel, and Others. Interpolation of animal tracking data in a fluid environment. *Journal of Experimental Biology*, 209(1):128–140, 2006.

[17] U. Turdukulov, A. O. Calderon Romero, O. Huisman, and V. Retsios. Visual mining of moving flock patterns in large spatio-temporal data sets using a frequent pattern approach. *International Journal of Geographical Information Science*, 28(10):2013–2029, 2014.

[18] M. R. Vieira, P. Bakalov, and V. J. Tsotras. On-line discovery of flock patterns in spatio-temporal data. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 286, New York, Nov. 2009. ACM Press.

[19] M. Wachowicz, R. Ong, C. Renso, and M. Nanni. Finding moving flock patterns among pedestrians through collective coherence. *International Journal of Geographical Information Science*, 25(11):1849–1864, 2011.

[20] E. A. Wentz, A. F. Campbell, and R. Houston. A comparison of two methods to create tracks of moving objects: linear weighted distance and constrained random walk. *International Journal of Geographical Information Science*, 17(7):623–645, 2003.

[21] K. Zheng, Y. Zheng, N. J. Yuan, S. Shang, and X. Zhou. Online discovery of gathering patterns over trajectories. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1974–1988, 2014.

[22] Y. Zheng and X. Zhou. *Computing with spatial trajectories*. Springer Science & Business Media, 2011.