Inclusão de Técnicas de Interpolação de Pontos em Algoritmos de Descoberta *On-Line* do Padrão *Flock*

Vitor Hugo Bezerra[1], **Daniel dos Santos Kaster**[1]

[1]Departamento de Computação – Universidade Estadual de Londrina (UEL)
Caixa Postal 10.011 – CEP 86057-970 – Londrina – PR – Brasil

vitorbezera@gmail.com, dskaster@uel.br

Abstract. *With decreasing costs of tracking devices and their availability in vehicles, smartphones and other devices, there is an increase of spatio-temporal data, which can be mined to find patterns regarding groups of moving objects. Among such patterns is the flock pattern that can be defined as a minimal number of entities within a defined distance radius moving together during a certain time-window. However, collection of object positions is usually irregular due to problems such as system failures or passing through tunnels or underground, resulting in gaps in trajectories that may preclude detection of moving patterns. One technique to address this problem is path interpolation, which geometrically generates the missing spatio-temporal points using a given estimation method. The objective of this work is the implementation and evaluation of the inclusion of interpolation techniques for the treatment of entries for on-line flock pattern algorithms. We present experiments that showed good results on finding flock patterns by applying interpolation compared to the results of utilizing the original datasets.*

Resumo. *Com a redução de custo e a maior disponibilidade de dispositivos de localização em veículos, smartphones e outros aparelhos, há um aumento de dados espaço-temporais, que podem ser minerados a fim de se encontrar padrões em grupos de objetos em movimento. Entre esses padrões está o padrão flock, que pode ser definido como um número mínimo de entidades dentro de um espaço delimitado por uma circunferência de raio definido que se deslocam juntos por um certo intervalo de tempo. No entanto, a coleta de posições de objetos é usualmente irregular devido a problemas, como falha de sistema ou falha por passagem em túneis, resultando em perdas nas trajetórias coletadas que podem impedir a identificação de padrões de movimentação. Uma solução para este problema é a interpolação de pontos que calcula geometricamente os pontos faltantes utilizando algum método de estimativa. O objetivo deste trabalho é a implementação e avaliação da inclusão de técnicas de interpolação para o tratamento de entradas para algoritmos de descoberta on-line do padrão flock. São apresentados experimentos que mostraram bons resultados na busca por padrões flock aplicando-se interpolação quando comparados aos resultados utilizando as bases de dados originais.*

1. Introdução

A crescente quantidade de dados espaço-temporais adquiridos atualmente tem ressaltado a necessidade de algoritmos para interpretar esses dados armazenados em grandes bancos de dados. A análise desse tipo de dado complexo, embora seja uma operação cara

computacionalmente, pode identificar comportamentos similares entre objetos como, por exemplo, migração de animais, detecção de congestionamentos em vias e de lugares com grande movimentação. Dentre as várias formas de análise de dados espaço-temporais, estão as de identificação de padrões de grupos de objetos em movimentando, mais especificamente aquele com representação em disco, que explora a distância máxima entre um número mínimo de entidades quaisquer não excedendo um diâmetro de disco definido em uma busca. Um dos padrões mais representativos da categoria de discos é o *flock*, que compreende um conjunto mínimo de objetos que estão espacialmente próximos por um intervalo de tempo pré-definido. Dados um número mínimo de entidades μ, um diâmetro de distância 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 tInterpolacaoempo consecutivos respeitando o espaço definido por um disco de diâmetro ϵ em cada instante de tempo [Vieira et al. 2009].

Entretanto, assim como no recebimento quanto na própria coleta desses dados de trajetórias, podem ocorrer falhas e ruídos, acarretando em atrasos e até mesmo perda de certas localizações coletadas. Essa possível irregularidade é causada por problemas como falhas em sistemas e dispositivos, por passagens por túneis ou subterrâneos, pela própria diferenciação das taxas de amostragem dos pontos de localização entre os diversos sistemas de coleta, etc. Como consequência desses possíveis *gaps* nas trajetórias, a identificação dos padrões *flock* pode ficar comprometida, devido à restrição temporal sequencial do padrão. Na prática, isto pode 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. Para o tratamento das trajetórias problemáticas, uma das técnicas que pode ser utilizada é a interpolação de pontos. Esta consiste em, a partir de determinados pontos coletados, interpolar e inserir pontos nos *gaps* dessas trajetórias.

Neste contexto, o objetivo deste trabalho é tratar o problema de detecção de *flocks* em trajetórias sujeitas a dados com variação de taxa de amostragem e *gaps* por falhas e/ou ruídos na coleta. A proposta é inserir técnicas de interpolação de pontos em algoritmos para a detecção on-line do padrão *flock*, possibilitando o tratamento de *streams* de dados de trajetórias de objetos móveis. Nossa proposta armazena dados recebidos em instantes subsequentes de tempo, guardando-os em estruturas de dados (*buffers*), detecta a falta de uma determinada posição e realiza a interpolação, gerando uma estimada para o objeto. 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. 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 respostas erradas, ou seja, falso-positivos.

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. Um padrão importante, foco deste trabalho, é o padrão *flock* (*flock pattern*). A definição mais utilizada na literatura define o padrão *flock* como um grupo de pelo menos m entidades movendo-se respeitando uma área máxima de um disco de diâmetro ϵ por um intervalo k de tempo [Vieira et al. 2009, Benkert et al. 2008, Arimura and Takagi 2014, Gudmundsson and van Kreveld 2006, Tanaka et al. 2015]. Formalmente, segundo [Vieira et al. 2009], um *flock* é dado pela 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 e 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, ..., $t_{j+\delta-1}$ em que existe um disco com centro $c_k^{t_i}$ e diâmetro ϵ cobrindo todos os pontos das trajetórias de $f_k^{t_i}$, que é flock f_k no instante t_i, $j \leq i \leq j + \delta$.*

Dentre os algoritmos para a identificação do padrão *flock*, dois merecem destaque por permitirem a detecção *on-line* do padrão: o BFE e o PSI. O algoritmo BFE (***B**asic **F**lock **E**valuation*) foi proposto por [Vieira et al. 2009] e utiliza um índice baseado em grade para processar os dados das trajetórias em cada instante de tempo. Já o algoritmo PSI (***P**lane Sweeping, Binary **S**ignatures and Inverted **I**ndex*), baseado no BFE e proposto por [Tanaka et al. 2015, Tanaka 2016], utiliza técnicas e estruturas de dados para detectar *flocks* de forma mais rápida que o BFE, como varredura de plano ao invés do índice de grade, assinatura binária e índice invertido. Nos testes apresentados pelos autores, o PSI apresentou desempenho superior ao BFE em vários conjuntos de dados reais para a maior parte das combinações de parâmetros do padrão. O mesmo aconteceu para a maior parte das variações dos algoritmos – heurísticas –, cujos detalhes podem ser encontrados nas referências originais ou em [Tanaka 2016].

2.2. Técnicas de Interpolação

Dados de trajetórias de objetos móveis podem ser incertos e incompletos, ou seja, imprecisos por 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 possa ser guardada em um certo período de tempo caso não haja a sincronização no tempo limite permitido [Zheng and Zhou 2011, Parent et al. 2013].

Na literatura há várias técnicas para o tratamento desses dados, incluindo a utilização de técnicas de *range queries* para conseguir os dados ideais para serem utilizados [Zheng and Zhou 2011], o que necessita uso de uma aplicação de banco de dados; aumentar o tempo entre um ponto e outro retirando alguns pontos, criando uma grande alteração no conjunto de dados pela falta de alguns pontos; utilização de filtros como o *Kalman Filter* [Gustafsson et al. 2002], que necessitam guardar uma grande quantidade de dados para realizar um funcionamento com bons resultados 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 [Li and Revesz 2002]. 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 informações semânticas. São eles:

- **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**: 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). 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 [Wentz et al. 2003].
- **Interpolação pela Curva de Bézier**: Já 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 [Tremblay et al. 2006].

3. Inclusão de Interpolação em Algoritmos *On-line* do Padrão *Flock*

Para os algoritmos de *flock*, incertezas nos dados de entrada são bastante prejudiciais. Dentre essas, destaca-se a perda de pontos de localização coletados, sendo mais impactante aos algoritmos *on-line*. Essa lacuna, ou *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 de *Flock* 1. Dessa forma, como impacto, os algoritmos podem identificar *flocks* com menos objetos que o na realidade, devido ao descarte destes com trajetórias com *gaps*, ou até mesmo deixar de reportar outros *flocks*, caso a remoção desses objetos descarte possível 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 e o PSI, utilizados neste trabalho.

Esta seção apresenta uma análise do comportamento desses algoritmos *on-line* de detecção do padrão *flock* quando as trajetórias dos objetos em estudo não têm pontos reportados em todos os seus instantes de tempo. Em seguida, é apresentada a proposta de inclusão de técnicas de interpolação nesses algoritmos, bem como os resultados de experimentos que confirmam a eficácia da proposta.

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 falta de pontos em um conjunto de dados e ainda permitir a comparação das respostas obtidas mediante diferentes situações de falta de dados, optou-se por fazer retirada controlada de pontos do conjunto de dados em teste. Desta forma, as respostas obtidas utilizando-se o conjunto de dados original 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 quesito. O método de retirada controlada de pontos foi desenvolvido em Python e trata separadamente as trajetórias de cada objeto do conjunto de dados. Para cada trajetória, é feita a contagem dos pontos seguida da retirada aleatória de uma certa porcentagem dos pontos desta, fornecida como parâmetro. O intuito dessa abordagem é garantir que a retirada de pontos seja homogênea 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 a comparação das saídas dos algoritmos, o método desenvolvido inicialmente faz uma limpeza dos dados e guarda os *flocks* reportados (conjunto de objetos que formam cada *flock* e intervalo de tempo) em listas ordenadas pelo instante de início do intervalo de tempo em que o *flock* foi identificado. Em seguida, é feita a comparação dos *flocks* reportados em cada arquivo, ou seja, em cada execução com percentual específico de retirada de pontos, indicando as taxas de *falso-negativos* (i.e., *flocks* que deveriam ter sido reportados, mas não foram) e de *falso-positivos* (i.e., *flocks* que não deveriam ter sido reportados, mas foram), para cada combinação de valores dos parâmetros dos algoritmos de detecção de *flocks* (μ, ϵ e δ).

A análise utilizou os conjuntos de dados originais dos trabalhos que propuseram os algoritmos BFE e PSI, bem como suas variações (vide Seção 2.1). A seguir são apresentados os resultados obtidos sobre os conjuntos de dados *Buses* e *Cars*. *Buses* contém 66.096 posições de ônibus se movendo na cidade de Atenas, Grécia. Já o conjunto *Cars* é composto de 134.264 posições coletadas a partir da movimentação de 183 carros privados movimentando-se em Copenhague, Dinamarca[1]. 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.

Conforme esperado, os resultados obtidos pelo BFE e o PSI foram idênticos em todos os casos, portanto, não será feita referência ao algoritmo na análise dos resultados que segue, apenas a qual conjunto de dados. A Figura 1 mostra que os algoritmos são muito sensíveis a faltas de pontos, pois mesmo a falta de 1% dos pontos faz com que

[1] www.daisy.aau.dk

os algoritmos deixem 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 aumentam a probabilidade de um ponto faltante dissolver um *flock* inteiro. Já o parâmetro ϵ apresentou uma variação menos acentuada de perda de respostas com o crescimento do valor do parâmetro.

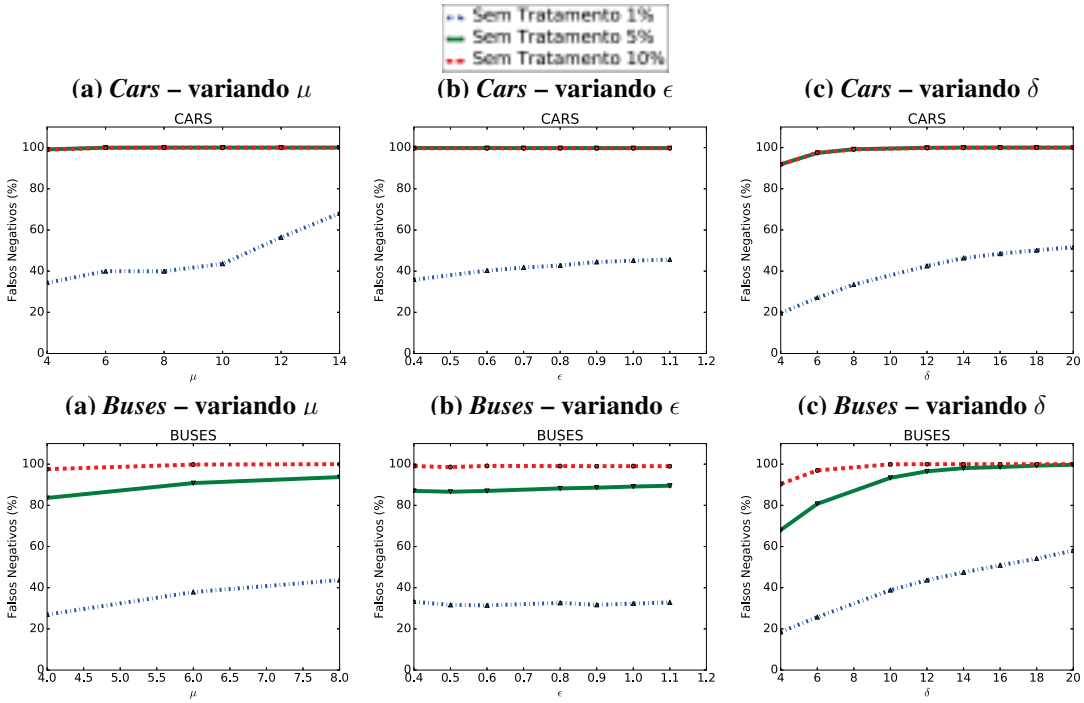

Figura 1. Taxa de falso-negativos gerado pelos algoritmos BFE e PSI mediante diferentes taxas de perdas/retirada de pontos. *Cars* (1ª linha) e *Buses* (2ª linha) variando μ (cardinalidade – 1ª coluna), ϵ (diâmetro dos discos – 2ª coluna) e δ (duração – 3ª coluna).

3.2. Proposta de Inclusão de Interpolação nos Algoritmos *On-line* de *Flocks*

Como visto na seção anterior, a falta 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 dos algoritmos BFE e PSI, 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 que poderá receber um ponto interpolado. Os demais *buffers* são utilizados para:

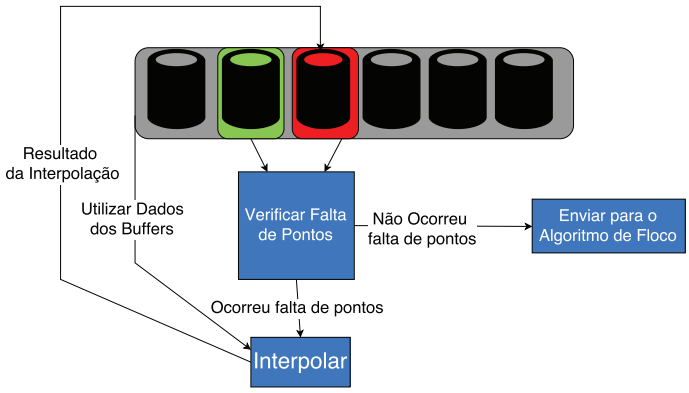

Figura 2. Exemplo do método proposto com a utilização de seis *buffers*.

(a) **identificar a perda de uma posição**: neste aspecto, assume-se que se uma trajetória continha pontos antes e depois do instante de tempo atual, então a falta do ponto no instante atual foi decorrência de uma falha de captura/transmissão; e
(b) **gerar um ponto interpolado**: os pontos nos instantes precedente(s) e subsequente(s) é(são) utilizado(s) para estimar o posicionamento do objeto no instante de tempo em que ocorreu a falha.

Desses *buffers* um é escolhido e os objetos contidos no *buffer* atual são comparados com os do *buffer* $t-1$. Se houver a falta de um objeto, então ele é interpolado com base nos dados desse objeto nos *buffers* de tempos anteriores e/ou futuros. Após a análise deste *buffer*, ele é liberado e os dados seguem para o algoritmo de detecção de *flock*. Se ao tentar interpolar um objeto ele não estiver em nenhum *buffer* futuro, não será interpolado o ponto, o que também funciona como método de parada para a interpolação, assim não ocorrendo interpolações depois do último tempo em que o objeto foi reportado.

Um exemplo do método pode ser visto na Figura 2, em que é ilustrado o método proposto com a utilização de seis *buffers*, representados na figura pelos cilindros. Na imagem, o *buffer* em vermelho 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* anterior (verde), 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, o ponto será interpolado com base nas posições do objeto contidos nos demais *buffers*.

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. Por esse motivo, na figura, há dois *buffers* anteriores ao *buffer* atual. Contudo, o número de *buffers* posteriores é maior devido ao fato que pode ter sido perdida a posição de um objeto em um instante (atual) e também em algum instante logo em seguida (por exemplo, no instante $t+2$). Este tipo de situação inviabilizaria a possibilidade de interpolação pela curva de Bézier. Depois de interpolar as posições necessárias, o *buffer* atual é liberado para o algoritmo de detecção do padrão *flock* processar o instante correspondente.

Vale ressaltar que a abordagem proposta insere um atraso na identificação do padrão *flock*, que depende do número de *buffers* posteriores ao instante atual. No en-

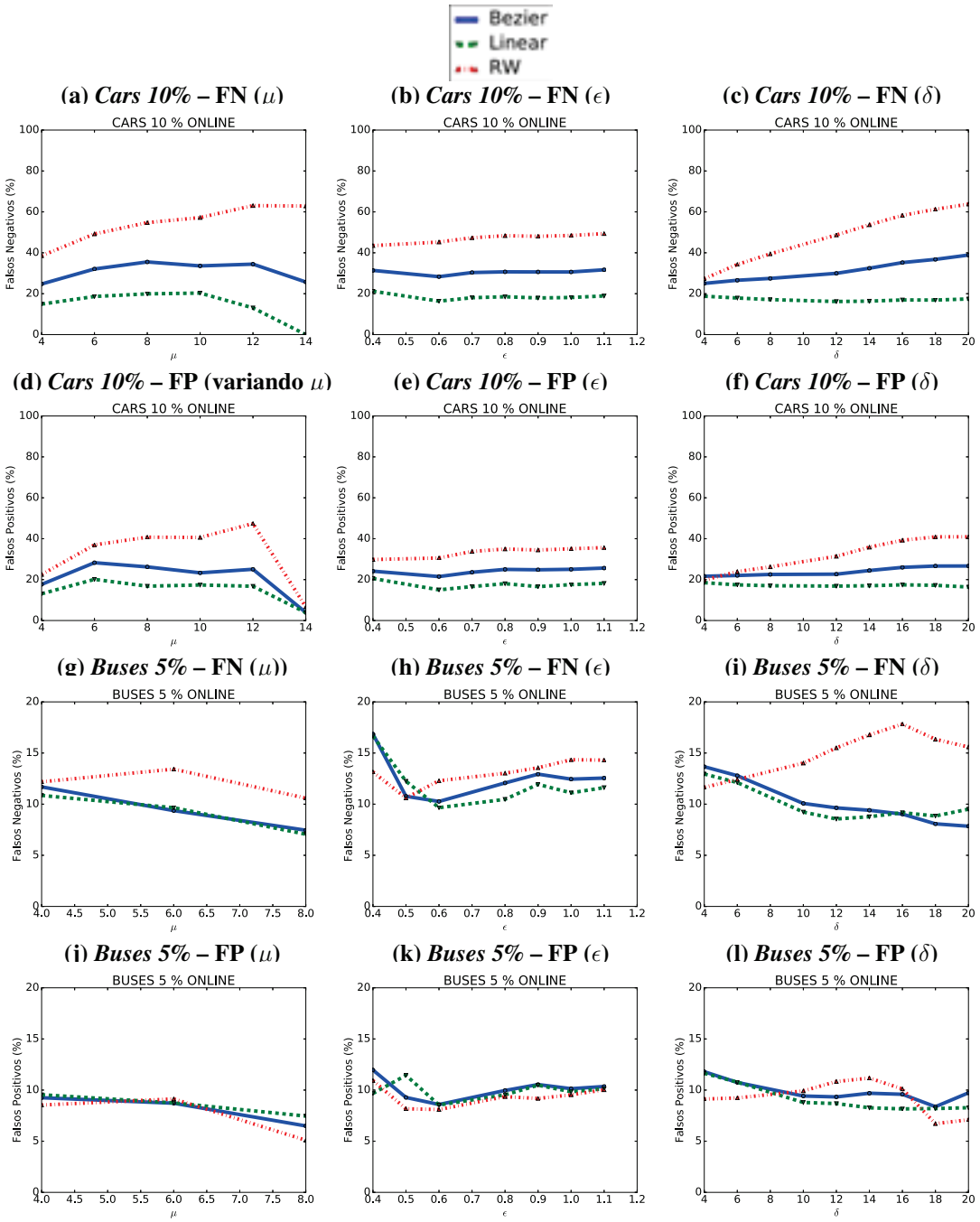

Figura 3. Taxa de acerto do método de interpolação *on-line*, variando μ (cardinalidade – 1ª coluna), ϵ (diâmetro dos discos – 2ª coluna) e δ (duração – 3ª coluna). *Cars* com 10% de perda – falso-negativos (1ª linha) e falso-positivos (2ª linha). *Buses* com 5% de perda – falso-negativos (3ª linha) e falso-positivos (4ª linha).

tanto, considera-se nesta proposta que o atraso é pequeno o suficiente para a maioria das aplicações que dependem da identificação *on-line* de *flocks*. Nestes casos, os benefícios do ganho de qualidade nos resultados sobrepõem o atraso inserido.

4. Avaliação Experimental da Proposta

Esta seção descreve os resultados obtidos sobre os conjuntos de dados *Cars* e *Buses*, avaliando a eficácia 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 Figura 3 mostra as taxas de acerto da proposta considerando 10% de perda no conjunto *Cars* e 5% no conjunto *Buses* para 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 para os dois conjuntos (figuras 3(a–c) e 3(g–i)), com destaque para a interpolação linear.

Esta inclusive possibilitou a redução da taxa de falso-negativos de quase 100% no conjunto *Cars* para pouco menos de 20% e a taxa do conjunto *Buses* de mais de 80% para menos de 10%. A curva de Bézier teve resultados pouco menos precisos, parte devido à natureza dos conjuntos de dados considerados, baseados em malha viária. Já a interpolação por *random walk* restrito apresentou o pior desempenho dentre as técnicas, em todos os casos, pois sua interpolação interfere muito na localização dos objetos criando *flocks* diferentes ou retirando objetos de respostas existentes . *Flocks* com menor tempo de duração e maior quantidade de objetos foram os que tiveram melhores recuperações de dados, já a diferença do tamanho do diâmetro do *flock* não resultou em grandes diferenças.

Observe-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 interpolados que diferem dos pontos de localização originais. Os gráficos das figuras 3(d–f) e 3(j–l) mostram que os falso-positivos fizeram a taxa total de erros dobrar, na maioria dos casos. Mas há casos em que os falsos positivos são respostas em que se diferenciam das originais pela falta ou adição de um elemento, o que também poderia ser considerado um floco original, o que uma análise menos sensível de diferença de respostas poderia mostrar. Entretanto, ainda usando uma análise sensível a diferenças, a taxa de erro global ainda é consideravelmente inferior do que a que obteve-se sem interpolação, sustentando assim a eficácia da proposta.

5. Conclusão

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*. Para abordar esse problema, este trabalho avaliou e desenvolveu formas para o tratamento de falta de pontos em trajetórias por meio de técnicas de interpolação de pontos. Foram testados três tipos de interpolação: Linear, *Constrained Random Walk* e Bezier, por obterem melhores resultados de acordo com a literatura. Foi desenvolvido um método de interpolação *on-line*, focado no recebimento de *strems* de dados. Nessa estratégia, como há um número limitado de dados para se realizar as interpolações, também são utilizados os pontos interpolados para a criação de novos pontos.

Também foram realizados diversos experimentos para avaliar o impacto da perda de dados de posicionamento e a eficácia da proposta. Os resultados mostraram que os algoritmos são muito sensíveis visto que uma pequena perda nos pontos já impacta significativamente. Observou-se, também, que quando as variáveis do padrão *flock* são muito

específicas, ou seja, valores pequenos de quantidade de objetos, intervalo de tempo e raio do círculo, a perda de *flocks* tende a ser menor. Os resultados obtidos com a aplicação da proposta possibilitaram um aumento expressivo na qualidade dos resultados, em particular quando foi utilizada a interpolação linear. Trabalhos futuros incluem a aplicação de outras técnicas de interpolação, utilizando técnicas que não dependam de dados futuros, e a avaliação de outras variáveis, como o impacto da proposta no tempo total de execução dos algoritmos e no consumo de memória.

Referências

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

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

[Gudmundsson and van Kreveld 2006] Gudmundsson, J. and van Kreveld, M. (2006). Computing longest duration flocks in trajectory data. In *Proceedings of the 14th ACM GIS*, page 35, New York, New York, USA. ACM Press.

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

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

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

[Tanaka 2016] Tanaka, P. S. (2016). *Algoritmos eficientes para detecção do padrão Floco em banco de dados de trajetórias*. Mestrado, Universidade Estadual de Londrina.

[Tanaka et al. 2015] Tanaka, P. S., Vieira, M. R., and Kaster, D. S. (2015). Efficient algorithms to discover flock patterns in trajectories. In *XVI Brazilian Symposium on Geoinformatics (GEOINFO)*, volume 1, pages 56–67, São Paulo, Brazil.

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

[Vieira et al. 2009] Vieira, M. R., Bakalov, P., and Tsotras, V. J. (2009). On-line discovery of flock patterns in spatio-temporal data. In *Proceedings of the 17th ACM GIS*, pages 286–295, New York, New York, USA. ACM Press.

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

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