UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ISABELLA STEFANNY FERNANDES DE ANDRADE

RetryTRACK: Recuperando Detecções Faltantes no Rastreamento de Pedestres Multi-câmera

Recife

2025

ISABELLA STEFANNY FERNANDES DE ANDRADE

RetryTRACK: Recuperando Detecções Faltantes no Rastreamento de Pedestres Multi-câmera

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Mídia e Interação

Orientador (a): Veronica Teichrieb

Coorientador (a): João Paulo Lima

Recife

2025

.Catalogação de Publicação na Fonte. UFPE - Biblioteca Central

Andrade, Isabella Stefanny Fernandes de.
RetryTRACK: recuperando detecções faltantes no rastreamento de pedestres multi-câmera / Isabella Stefanny Fernandes de Andrade. - Recife, 2024.
58f.: il.

Dissertação (Mestrado) - Universidade Federal de Pernambuco, Centro de Informática, Programa de Pós-graduação em Ciência da Computação, 2024.
Orientação: Veronica Teichrieb.
Coorientação: João Paulo Lima.
Inclui referências.

1. Rastreamento; 2. Multi-câmera; 3. Regressão de Processo Gaussiano. I. Teichrieb, Veronica. II. Lima, João Paulo. III. Título.

UFPE-Biblioteca Central

Isabella Stefanny Fernandes de Andrade

“RetryTRACK: Recuperando Detecções Faltantes no Rastreamento de Pedestres Multi-câmera”

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Inteligência Computacional.

Aprovado em: 01/11/2024.

_____

Orientadora: Profa. Dra. Veronica Teichrieb

BANCA EXAMINADORA

_____

Prof. Dr. Silvio de Barros Melo
Centro de Informática / UFPE

_____

Prof. Dr. David Menotti Gomes
Departamento de Informática / UFPR

_____

Prof. Dr. João Paulo Silva do Monte Lima
Departamento de Computação / UFRPE
(coorientador)

Dedico aos meus pais e aos meus amigos.

AGRADECIMENTOS

Primeiramente, agradeço aos meus pais Kátia e Arlindo Júnior, que sempre me apoiaram em minhas escolhas, ofereceram ajuda, e proporcionaram boas condições de vida.

Em segundo, agradeço aos meus amigos, Clara e Yves, pelos inúmeros momentos inesquecíveis, como nossa viagem a São Paulo. Graças a vocês, pude me distrair e aproveitar um pouco a vida.

Agradeço também a meus colegas do projeto SABIÁ como Pedro, Júlia, Amadeo e Ariany, com os quais pude colaborar e aprender coisas novas, e aos meus colegas do Voxar Labs, como Isabela, Malu, Will e André, que me acompanharam durante a aventura no CVPR. Agradeço ainda ao professor Rafael, que me auxiliou em vários momentos durante a minha pesquisa, e a Dani, que me ajudou a me organizar e a cumprir metas para chegar ao fim deste trabalho. Agradeço a todos do Voxar que fizeram sugestões ao longo da minha pesquisa, a infraestrutura do Voxar foi essencial para meu crescimento como pesquisadora. Obrigada a Maria também, que exerce um papel fundamental para o laboratório funcionar, além de se preocupar em criar uma boa atmosfera para todos.

Agradeço a minha orientadora Vt, que tenta incentivar todos do Voxar a irem além, e está sempre buscando oportunidades novas com parceiros. E agradeço ao meu co-orientador Jonga, que sempre foi um excelente orientador, que esteve sempre disposto a ajudar, e é também um excelente pesquisador, que se mantém atualizado e consegue contribuir para pesquisas impactantes no estado-da-arte. Obrigada por ter sido compreensivo ao longo do meu mestrado.

RESUMO

O rastreamento de pedestres costuma depender de algoritmos de detecção, porém esses algoritmos nem sempre estão corretos e podem perder alguns pedestres. Embora usar várias câmeras seja uma maneira de lidar com isso, algumas falhas ainda ocorrem. Por conta disso, é desejável que o rastreador tente corrigir as detecções faltantes utilizando as informações temporais. Este trabalho apresenta um método online que não requer treinamento para recuperar detecções ausentes durante o rastreamento de várias câmeras. A técnica utiliza extrapolação linear e regressão de processo gaussiano para produzir novas coordenadas suavizadas. Além disso, propomos um filtro para remover detecções duplicadas. Anexamos o módulo a um rastreador de múltiplas câmeras base e o avaliamos nas bases de dados WILDTRACK e MultiviewX. A precisão de rastreamento foi melhorada em 0,95 p.p. no WILDTRACK e 2,34 p.p. no MultiviewX com a adição do módulo. Além disso, essa estratégia recuperou com sucesso 17,98% das detecções ausentes no WILDTRACK e significativos 40,12% no MultiviewX, ressaltando sua aplicação prática e potencial para resolver o problema.

Palavras-chaves: Rastreamento. Multi-câmera. Regressão de Processo Gaussiano.

ABSTRACT

Tracking pedestrians commonly relies on detection algorithms. However, these algorithms are not always correct and may miss some pedestrians. Although using multiple cameras is a way to handle this, some failures still need to be fixed. Thus, it is desirable that the tracker attempt to fix the detections. This work introduces an online and unsupervised module designed to recover missing detections during multiple camera tracking. The module applies linear extrapolation and Gaussian process regression techniques to produce new smoothed coordinates. Furthermore, we propose a filter to remove duplicate detections. We attached the module to a multi-camera baseline tracker and evaluated it on the WILDTRACK and MultiviewX datasets. The multiple object tracking accuracy was improved by 0.95 p.p. on WILDTRACK and 2.34 p.p. on MultiviewX with the addition of the module. Moreover, this strategy successfully recovered 17.98% of missing detections in WILDTRACK and a significant 40.12% in MultiviewX, underscoring its practical application and potential to address the problem.

Keywords: Tracking. Multi-camera. Gaussian Process Regression.

LISTA DE FIGURAS

Figura 1 – Ilustração de como funciona o rastreamento de pedestres. Os pedestres #1, #2 e #3 são localizados e re-identificados em diferentes quadros. 16

Figura 2 – Exemplo de aplicação do rastreamento de pedestres no varejo, onde os dados do rastreamento são utilizados para auxiliar o gerenciamento de filas. 16

Figura 3 – Exemplo de aplicação do rastreamento de pedestres na interação humano-máquina, onde o drone possui um sistema de rastreamento para poder acompanhar a pessoa. 17

Figura 4 – Exemplo de aplicação de vigilância em tempo real, onde as pessoas são re-identificadas ao longo do tempo e um sistema de busca por palavras-chave permite filtrar indivíduos por suas características. 17

Figura 5 – Casos de falha após utilizar o algoritmo de detecção de Lima et al. (2021). Os erros são adicionados ao comparar os resultados de detecção com as anotações verdadeiras. 18

Figura 6 – Ilustração dos eixos de translação e rotação em relação à câmera. 21

Figura 7 – Demonstração da imagem da câmera utilizando diferentes valores de distância focal. Ao aumentar a distância focal, os objetos aparecem mais próximos, porém, o campo de visão é menor. 22

Figura 8 – Ilustração demonstrando o ponto principal, que é a interseção do eixo principal com o plano de projeção que passa pelo ponto focal. 22

Figura 9 – Detecção de pedestres utilizando HOG de Dalal e Triggs (2005). Da esquerda para a direita, a primeira imagem mostra o gradiente médio das imagens de treino. A segunda imagem mostra em branco quais locais possuem os pesos positivos na SVM (que são mais relevantes para realizar a classificação), enquanto a terceira imagem enfatiza os pesos negativos. A quarta imagem é uma imagem de teste, e a quinta imagem é o seu HOG computado. A sexta e a sétima imagem são o HOG ponderado pelos pesos positivos e negativos respectivamente. 23

Figura 10 – Detecção de pedestres utilizando HOG baseado em partes de Felzenszwalb, McAllester e Ramanan (2008). Da esquerda para a direita, um pedestre é detectado (caixa azul) e suas partes são identificadas (caixas amarelas). Cada parte possui um HOG de alta dimensionalidade. Uma pessoa é detectada corretamente, mas uma vaca também é identificada como pedestre incorretamente. 24
Figura 11 – Detecção de objetos de Liu et al. (2016). A primeira imagem contém a anotação verdadeira de duas caixas delimitadoras envolvendo animais. O método SSD considera mapas de diferentes escalas (nesse caso, um de 8x8 na segunda imagem e um de 4x4 na terceira imagem), e em cada localização dentro do mapa considera as caixas delimitadoras pré-definidas (quatro nesse exemplo) e tenta inferir a localização e a confiança para cada classe. 25
Figura 12 – Detecção de objetos utilizando YOLO (REDMON et al., 2016). Cada célula ajuda a inferir caixas delimitadoras de diferentes tamanhos junto com um nível de confiança. 25
Figura 13 – Exemplo de visualização das trajetórias dos pedestres obtidas após o rastreamento. 26
Figura 14 – Processamento do rastreamento *offline*. As etapas devem ser seguidas de forma linear, onde primeiro é necessário receber todas as imagens da sequência, depois o rastreamento é feito sobre toda a sequência e por fim os resultados são obtidos. 28
Figura 15 – Processamento do rastreamento *online*. As etapas ocorrem de forma iterativa, onde a cada instante t o rastreamento é feito considerando as detecções que temos até o momento e já é possível utilizar os resultados. 28
Figura 16 – Demonstração do ByteTrack (ZHANG et al., 2022). O rastreador recebe as detecções e suas taxas de confiança. Em seguida, associa primeiro as detecções com alta taxa de confiança, e depois associa as restantes. 30

Figura 17 – O método BoostTrack (STANOJEVIC; TODOROVIC, 2024) propõe que a taxa de confiança de detecções que tem a posição similar à posição prevista pelo filtro de Kalman seja aumentada. Além disso, uma taxa de confiança que relaciona detecções e trajetórias é proposta para ser utilizada em conjunto com matrizes de similaridade base que já foram propostas em outros trabalhos. 31
Figura 18 – Comparação da trajetória dos pedestres nas anotações verdadeiras (verde) com a trajetória dos pedestres usando diferentes módulos do rastreamento. Inicialmente, o rastreamento ocorre com bastante ruído e mudanças bruscas (laranja), além de deixar de ocorrer durante oclusões. A interpolação linear (azul) recupera o pedestre durante a oclusão. E incluindo a função gaussiana (vermelho) a trajetória rastreada se torna mais suave e próxima das anotações verdadeiras. 31
Figura 19 – Algoritmo de Nguyen et al. (2022). (a) As caixas delimitadoras são entrada para o método. (b) As coordenadas são projetadas no plano de chão 3D. (c) Um grafo espaço-temporal é criado com as detecções. (d) Os custos de associação entre os nós são computados utilizando redes neurais de grafos. (e) A associação global de trajetórias é feita. (f) As coordenadas 3D das trajetórias são geradas. 33
Figura 20 – Demonstração de de Vo et al. (2021). As detecções de aparência similar são agrupadas. 33
Figura 21 – Algoritmo de You e Jiang (2020). Uma rede neural estima o mapa de calor com as detecções em cada câmera. Em seguida, eles são projetados para o plano de chão. Eles utilizam uma rede neural para classificar se as detecções são positivas para remover falsos positivos, e realizam o rastreamento criando um grafo bipartido e fazendo o acoplamento. 34
Figura 22 – Algoritmo de Cheng et al. (2023). (a) Associação espaço-temporal de detecções utilizando redes neurais de grafos. (b) Arquitetura da rede neural de grafo. (c) Pós-processamento de trajetórias. (d) Agregamento de grafos temporais e espaciais. 35

Figura 23 – Algoritmo de Andrade e Lima (2023). (a) As caixas delimitadoras dos pedestres são entrada para o algoritmo. (b) Um detector 3D que devolve as detecções no plano de chão é utilizado. (c) Um grafo é criado conectando detecções de quadros diferentes. (d) A rede é treinada ao compartilhar informações entre nós vizinhos e arestas. (e) As arestas são classificadas como ativas ou inativas. (f) O algoritmo devolve as trajetórias no plano de chão. 35
Figura 24 – Processamento de Ong et al. (2022). O filtro de Bayes estima as trajetórias e um módulo auxiliar gerencia as oclusões. 36
Figura 25 – Algoritmo de Lyra. et al. (2022). Um grafo é criado com as detecções, e o algoritmo de acoplamento associa as detecções mais perto. A posição das detecções é atualizada com o filtro de Kalman. 36
Figura 26 – Imagens de múltiplas câmeras são entrada para o detector, que devolve as detecções como coordenadas de mundo no plano de chão. Um rastreador base calcula as trajetórias dos pedestres, e nossa extrapolação gaussiana suavizada refina os resultados, recuperando detecções ausentes. A saída são as trajetórias de pedestres atualizadas. 37
Figura 27 – Processo de detecção. Os pedestres são detectados em cada câmera, e em seguida o ponto no plano de chão é estimado a partir de cada câmera. Os pontos de chão de diferentes câmeras são agrupados considerando a distância euclidiana, e opcionalmente a distância entre os vetores de reidentificação. No final, obtemos a coordenada 3D do pedestre. 39
Figura 28 – Processo de rastreamento. A cada instante de tempo, o algoritmo recebe as imagens das múltiplas câmeras e utiliza o detector mencionado na Seção 4.1. Se estiver no primeiro instante de tempo, atribui identidades iniciais. Se não, monta o grafo e utiliza o algoritmo de acoplamento. Em seguida, atualiza o estado do objeto no filtro de Kalman, e retorna o resultado do rastreamento. Antes de realizar o próximo acoplamento, o filtro de Kalman é utilizado para prever a posição de pedestres e obter uma localização mais aproximada de onde eles estariam no quadro atual utilizando sua velocidade anteriormente. O processo se repete até chegar ao instante final da sequência. 40

Figura 29 – Ilustração das etapas do módulo GSE. Primeiro ocorre a extrapolação linear, que gera uma nova coordenada para pedestres que não foram identificados no tempo atual. Em seguida, as observações de quadros anteriores são usadas para treinar o modelo de GPR. Por fim, inferimos as coordenadas suavizadas. 41
Figura 30 – Imagem ilustrando um conjunto de pontos e múltiplas funções que se ajustam a eles. 42
Figura 31 – Exemplo entre funções antes e depois da aplicação da RBF. 42
Figura 32 – Exemplo de caso onde duas detecções estão muito próximas (#030 e #021). 43
Figura 33 – Imagens das sete câmeras da base de dados WILDTRACK. 44
Figura 34 – Imagens das seis câmeras da base de dados MultiviewX. 45
Figura 35 – Detecções de pedestres reais que não estão nas anotações WILDTRACK e foram classificados como FP. 48
Figura 36 – O pedestre #22 estava sendo rastreado inicialmente. No entanto, ele desapareceu no quadro seguinte. Posteriormente, o GSE conseguiu recuperar a detecção. 49
Figura 37 – Um pedestre que não é detectado em nosso método, apesar de estar visível, pois só aparece em uma câmera. 50

LISTA DE TABELAS

Tabela 1 – Comparação com outras técnicas de última geração no conjunto de dados WILDTRACK. Avaliamos usando os últimos 10% dos quadros para comparar com métodos supervisionados. Ong et al. (2022) é uma exceção que usou todo o conjunto de dados. O nosso é o melhor método não supervisionado. 47

Tabela 2 – Resultados de cada módulo, além do rastreador base no conjunto de dados WILDTRACK. Tentamos replicar o rastreador base e obtivemos um resultado semelhante, mas diferente. Incluímos ambos os resultados na tabela para esclarecimento. Adicionamos GSE e o filtro de duplicatas à replicação do rastreador base. 48

Tabela 3 – Resultados do rastreador proposto usando as detecções mais recentes de Lima et al. (2023) na base de dados WILDTRACK. 49

Tabela 4 – Resultados usando um número diferente de câmeras. 51

Tabela 5 – Resultados de cada módulo, além do rastreador base no conjunto de dados MultiviewX. 52

LISTA DE ABREVIATURAS E SIGLAS

AOI	Área de Interesse - *Area of Interest*
FN	Falso Negativo
FP	Falso Positivo
FPS	quadros por segundo - *frames per second*
GPR	Regressão de Processo Gaussiano - *Gaussian Process Regression*
GSE	Extrapolação com Suavização Gaussiana - *Gaussian-smoothed Extrapolation*
GSI	Interpolação com Suavização Gaussiana - *Gaussian-smoothed Interpolation*
HOG	Histograma de Gradientes Orientados - *Histogram of Oriented Gradients*
IOU	Interseção sobre União - *Intersection Over Union*
MM	Troca de Identidade - *Mismatch*
MOTA	Acurácia de Rastreamento de Múltiplos Objetos - *Multiple Object Tracking Accuracy*
MOTP	Precisão de Rastreamento de Múltiplos Objetos - *Multiple Object Tracking Precision*
Prec	Precisão - *Precision*
RBF	Função de Base Radial - *Radial Basis Function*
Rcll	Revocação - *Recall*
SSD	Detector Multicaixa de Disparo Único - *Single-shot Multibox Detector*
SVM	Máquina de Vetores de Suporte - *Support Vector Machine*
TBD	Rastreamento por Detecção - *Tracking-by-Detection*
VP	Verdadeiro Positivo
YOLO	You Only Look Once

SUMÁRIO

1 INTRODUÇÃO 16

1.1 OBJETIVOS DO TRABALHO 18

1.2 ESTRUTURA DA DISSERTAÇÃO 19

2 FUNDAMENTAÇÃO TEÓRICA 20

2.1 CALIBRAÇÃO DE CÂMERAS 20

2.2 DETECÇÃO DE PEDESTRES 23

2.3 RASTREAMENTO DE PEDESTRES 26

3 TRABALHOS RELACIONADOS 29

3.1 RASTREAMENTO COM CÂMERA ÚNICA 29

3.2 RASTREAMENTO COM MÚLTIPLAS CÂMERAS 32

4 RETRYTRACK 37

4.1 DETECTOR DE PEDESTRES MULTI-CÂMERA 37

4.2 RASTREADOR DE PEDESTRES MULTI-CÂMERA BASE 39

4.3 EXTRAPOLAÇÃO COM SUAVIZAÇÃO GAUSSIANA 40

4.4 FILTRO DE DETECÇÕES DUPLICADAS 43

5 EXPERIMENTOS 44

5.1 BASE DE DADOS 44

5.1.1 WILDTRACK 44

5.1.2 MultiviewX 45

5.2 MÉTRICAS 45

5.3 COMPARAÇÃO COM O ESTADO DA ARTE 46

5.4 ESTUDO DE ABLAÇÃO 47

5.4.1 Resultados no WILDTRACK 47

5.4.1.1 Variação no Número de Câmeras 50

5.4.2 Resultados no MultiviewX 51

6 CONSIDERAÇÕES FINAIS 53

6.1 CONCLUSÕES 53

6.2 CONTRIBUIÇÕES 53

6.3 TRABALHOS FUTUROS 54

REFERÊNCIAS 55

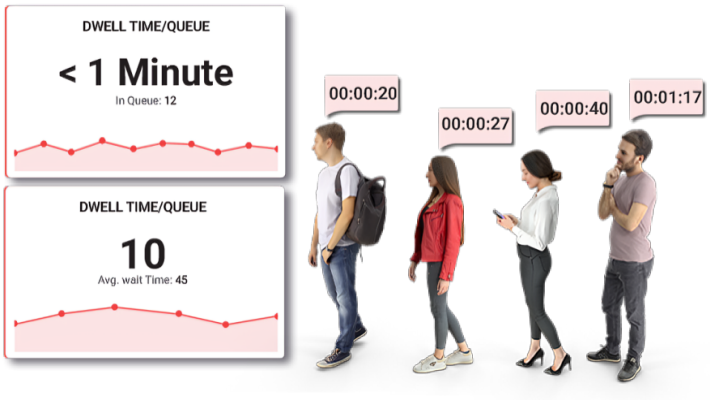
1 INTRODUÇÃO

O rastreamento de pedestres é uma tarefa importante que envolve localizar indivíduos ao longo do tempo (Figura 1). Ele tem aplicações em muitos setores, como varejo (HERNANDEZ; NALBACH; WERTH, 2019), interação humano-máquina (BEVILACQUA; MAIO, 2016) e reabilitação (ZHOU; HU, 2008). As Figuras 2, 3 e 4 ilustram algumas aplicações.

Figura 1 – Ilustração de como funciona o rastreamento de pedestres. Os pedestres #1, #2 e #3 são localizados e re-identificados em diferentes quadros.

Fonte: Elaborada pela autora (2024)

Figura 2 – Exemplo de aplicação do rastreamento de pedestres no varejo, onde os dados do rastreamento são utilizados para auxiliar o gerenciamento de filas.



Fonte: VEMCO GROUP (2024)[1]

Algumas aplicações, como por exemplo sistemas de vigilância em tempo real (LEI et al., 2022) e veículos autônomos (LU et al., 2022), se apoiam em métodos de rastreamento online para obter informação instantânea. Nesses métodos, não é possível incorporar informações de quadros futuros durante o rastreamento.

Além disso, parte dos algoritmos de rastreamento utiliza a posição 2D dos pedestres (BEWLEY et al., 2016; BERGMANN; MEINHARDT; LEAL-TAIXE, 2019; ZHOU; KOLTUN; KRÄHENBÜHL,

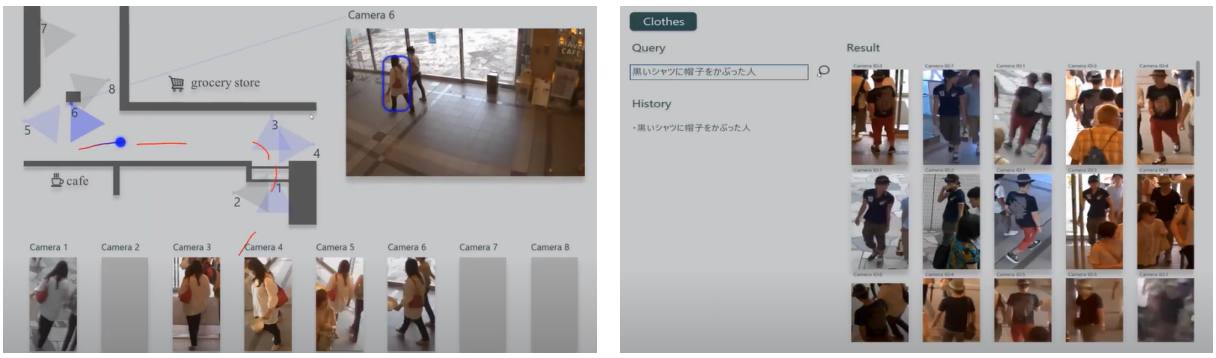
[1] <https://vemcogroup.com/pt/solucao/gerenciamento-de-filas>. Acessado em 10 de Julho de 2024.

Figura 3 – Exemplo de aplicação do rastreamento de pedestres na interação humano-máquina, onde o drone possui um sistema de rastreamento para poder acompanhar a pessoa.

Fonte: Adaptado de BEVILACQUA; MAIO (2016)

2020). Contudo, a posição 3D é essencial para entender melhor o ambiente e é necessária para tarefas de visão computacional como navegação de robôs (KIM; OŠEP; LEAL-TAIXé, 2021) e realidade aumentada (AHMADYAN et al., 2020).

Figura 4 – Exemplo de aplicação de vigilância em tempo real, onde as pessoas são re-identificadas ao longo do tempo e um sistema de busca por palavras-chave permite filtrar indivíduos por suas características.



Fonte: TOSHIBA (2017)[2]

Os algoritmos de rastreamento costumam utilizar o paradigma de *Rastreamento por Detecção* - Tracking-by-Detection *(TBD)*, como discutido no *survey* de Sun et al. (2020). Neste paradigma, primeiro um detector de objetos localiza os pedestres em cada quadro. Em seguida, o rastreador associa as detecções que pertencem à mesma pessoa, atribuindo a mesma identidade para elas.

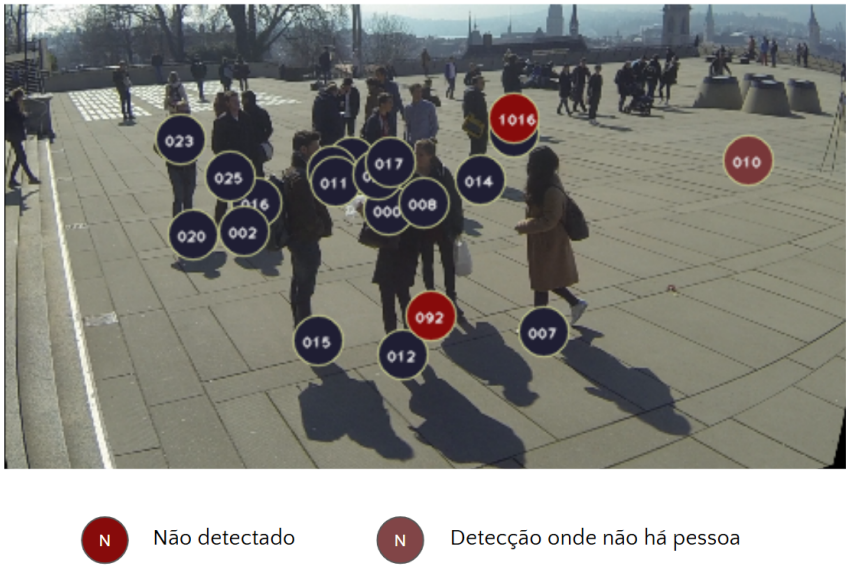
Portanto, a acurácia de rastreamento é altamente dependente da qualidade das detecções. Porém, em alguns casos o detector falha em detectar a presença de uma pessoa, por exemplo devido a oclusões quando outro pedestre ou um objeto bloqueia a visão. Nesse cenário, utilizar múltiplas câmeras é útil para recuperar pessoas que estão visíveis a partir de outro ponto de vista. (LIMA et al., 2021).

[2] <https://www.global.toshiba/ww/technology/corporate/rdc/rd/topics/17/1708-01.html>. Acessado em 10 de Julho de 2024.

Muitas técnicas de estado da arte (YOU; JIANG, 2020; VO et al., 2021; CHENG et al., 2023; NGUYEN et al., 2022) treinam redes neurais para rastrear em múltiplas câmeras. No entanto, métodos supervisionados requerem retreinamento em novos ambientes, impedindo aplicações no mundo real.

Ademais, mesmo com múltiplas câmeras, os algoritmos de detecção têm dificuldade para identificar todos os pedestres. A Figura 5 exibe alguns casos de falha que ocorrem ao utilizar o detector de Lima et al. (2021). Sendo assim, alguns métodos de rastreamento propuseram algoritmos que buscam tratar detecções faltantes (ZHANG et al., 2022; DU et al., 2023).

Figura 5 – Casos de falha após utilizar o algoritmo de detecção de Lima et al. (2021). Os erros são adicionados ao comparar os resultados de detecção com as anotações verdadeiras.



Fonte: Elaborada pela autora (2024)

1.1 OBJETIVOS DO TRABALHO

Neste trabalho, propomos combinar um rastreador 3D multi-câmera com um módulo inspirado no StrongSORT (DU et al., 2023) que recupera detecções faltantes. Nós adaptamos esse módulo para ser online e para funcionar em ambientes multi-câmera. Além disso, nós incluímos um filtro para remover detecções duplicadas. Nós aplicamos o módulo e o filtro ao rastreador 3D proposto por Lyra. et al. (2022), uma técnica de rastreamento multi-câmera de estado da arte. O rastreador 3D e o módulo são não supervisionados e adequados para aplicações práticas.

Portanto, ao longo dessa pesquisa, nós analisamos a seguinte hipótese:

- **H1:** O uso da extrapolação com suavização gaussiana contribui para recuperar falhas no rastreamento 3D online multi-câmera de pedestres.

Além disso, as contribuições desse trabalho são:

- Um módulo online e multi-câmera para recuperar detecções faltantes 3D que pode ser acoplado a rastreadores existentes (Capítulo 4);
- Avaliações quantitativas comparando com rastreadores multi-câmera 3D de estado da arte, e também avaliações qualitativas de erros que aconteciam antes da adição do módulo, erros que ainda acontecem mesmo com o módulo, e casos de sucesso (Capítulo 5).

1.2 ESTRUTURA DA DISSERTAÇÃO

Os próximos capítulos deste trabalho estão organizados da seguinte forma:

- Capítulo 2 - FUNDAMENTAÇÃO TEÓRICA: Explica conceitos-chave como calibração de câmeras, detecção de pedestres e rastreamento de pedestres.
- Capítulo 3 - TRABALHOS RELACIONADOS: Revisita outros trabalhos que realizam o rastreamento utilizando uma câmera ou múltiplas câmeras.
- Capítulo 4 - RETRYTRACK: Descreve o método aplicado para rastrear os pedestres, desde as detecções e rastreamento base, até o módulo incorporado e o filtro de detecções duplicadas.
- Capítulo 5 - EXPERIMENTOS: Apresenta e discute os resultados obtidos em experimentos.
- Capítulo 6 - CONSIDERAÇÕES FINAIS: Aborda as contribuições do trabalho, conclusões e sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, alguns conceitos são introduzidos para explicar desde a detecção de pedestres até o rastreamento. A Seção 2.1 define quais são os parâmetros de calibração de câmera, necessários para a detecção utilizada no método proposto. A Seção 2.2 revisita métodos clássicos e atuais de detecção de pedestres. E a Seção 2.3 explica o procedimento comum para realizar o rastreamento de pedestres.

2.1 CALIBRAÇÃO DE CÂMERAS

As imagens digitais são capturas dos pontos 3D do mundo projetados no plano 2D. É possível fazer tanto a projeção 3D para 2D quanto a desprojeção 2D para 3D se os parâmetros da câmera e a profundidade do ponto forem conhecidos. Existem dois tipos de parâmetros, os extrínsecos e os intrínsecos.

Os parâmetros extrínsecos descrevem a relação da câmera com o mundo. Por isso, eles são formados pela matriz de rotação $\mathbf{R}$ e pelo vetor de translação $\mathbf{t}$. A matriz $\mathbf{R}$ é definida por

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \tag{2.1}$$

onde cada coluna representa a rotação nos eixos da câmera x, y e z em torno de um eixo arbitrário.

Já o vetor $\mathbf{t}$ é definido por

$$\mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}, \tag{2.2}$$

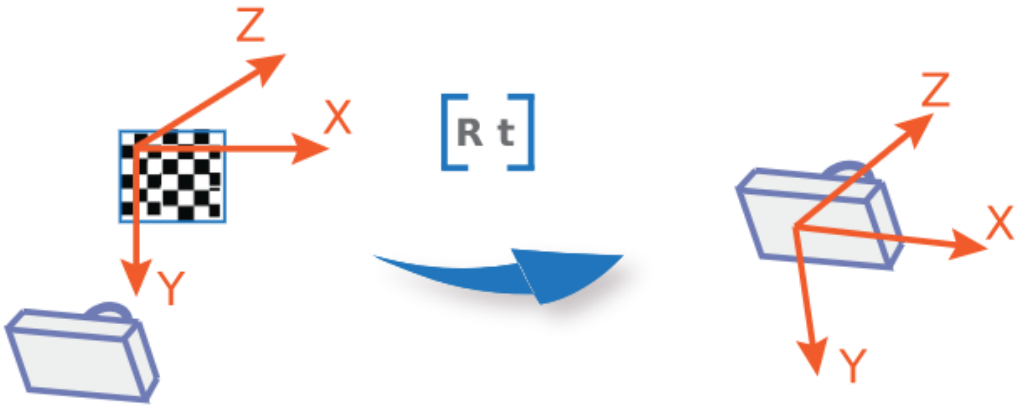
e é responsável por definir a posição do centro óptico da câmera em relação ao ponto de origem do sistema de coordenadas de mundo.

A matriz de rotação e o vetor de translação podem ser combinados e representados por

$$[\mathbf{R}|\mathbf{t}] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}. \tag{2.3}$$

A Figura 6 ilustra os parâmetros extrínsecos.

Figura 6 – Ilustração dos eixos de translação e rotação em relação à câmera.



Fonte: MathWorks (2024)[1]

Enquanto isso, os parâmetros intrínsecos são características únicas sobre como a câmera captura as imagens. Neles, está incluída a distância focal, que é a distância entre o ponto focal e o sensor de imagem, e quanto maior for, menor é o campo de visão e maior é a aproximação dos objetos. Ela é definida por

$$f = \alpha \times s_t \tag{2.4}$$

onde α é uma constante que relaciona o tamanho do sensor físico com o tamanho em pixels e s_t é o tamanho do sensor. A Figura 7 exibe imagens tiradas do mesmo local com diferentes valores de distância focal.

Outro componente dos parâmetros intrínsecos é o ponto principal $(c_x, c_y)^T$. O ponto principal é a interseção do eixo principal com o plano de projeção. O eixo principal é uma reta perpendicular ao plano de projeção que passa pelo ponto focal. Esse ponto é essencial para tarefas como alinhamento de imagem. Ele pode ser calculado por

$$c_x = \frac{W}{2} \tag{2.5}$$

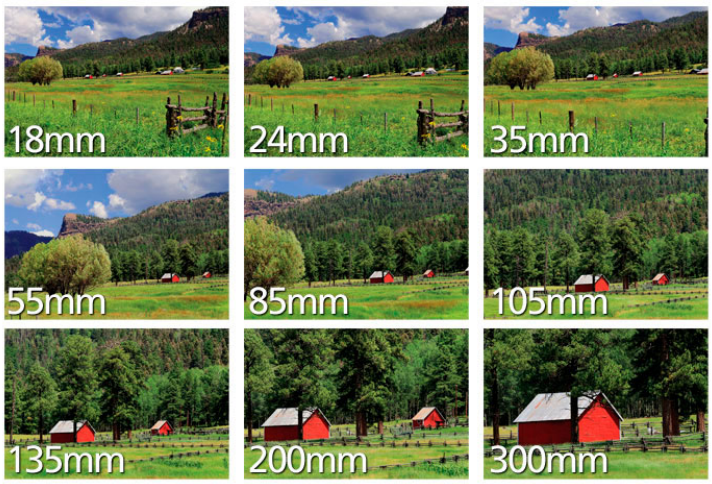
$$c_y = \frac{H}{2}, \tag{2.6}$$

onde W é a largura e H é a altura da imagem. A Figura 8 mostra o ponto principal em relação a outros pontos de referência da câmera.

Além disso, um parâmetro menos encontrado nas câmeras modernas é a distorção da lente s. É a descrição do ângulo entre os eixos x e y no plano da imagem e costuma ser zero.

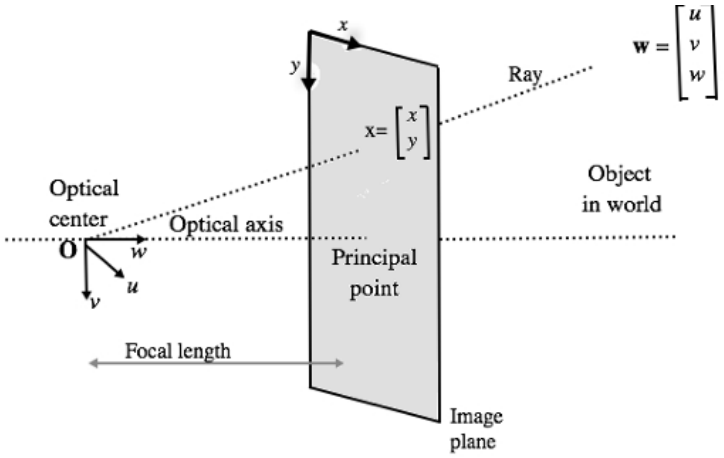
[1] <https://www.mathworks.com/help/vision/ug/camera-calibration.html>. Acessado em 10 de Julho de 2024.

Figura 7 – Demonstração da imagem da câmera utilizando diferentes valores de distância focal. Ao aumentar a distância focal, os objetos aparecem mais próximos, porém, o campo de visão é menor.



Fonte: BLACK (2024)[2]

Figura 8 – Ilustração demonstrando o ponto principal, que é a interseção do eixo principal com o plano de projeção que passa pelo ponto focal.



Fonte: AZARCOYA-CABIEDES et al. (2014)

Unindo os parâmetros definidos acima, a matriz intrínseca é então denotada por

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.7}$$

Por fim, a matriz de projeção, que é utilizada para mapear coordenadas de mundo 3D para a imagem 2D em coordenadas locais de imagem, é denotada por

$$\mathbf{P} = K[R|t]. \tag{2.8}$$

[2] <https://www.nikonusa.com/learn-and-explore/c/tips-and-techniques/understanding-focal-length>. Acessado em 10 de Julho de 2024.

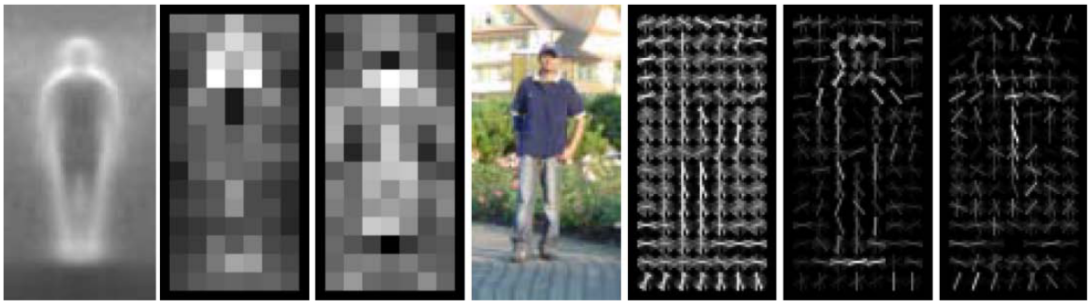
2.2 DETECÇÃO DE PEDESTRES

As imagens são representadas digitalmente como matrizes de números. O computador não compreende os seus elementos como nós humanos. Portanto, para extrair de forma automática a posição de pedestres, é necessário elaborar algoritmos que consigam interpretar as informações contidas nos números para uma linguagem mais humana.

Para descrever a posição do pedestre em uma imagem, normalmente se utiliza uma caixa delimitadora, que é o retângulo mínimo que envolve todos os pontos do pedestre. Esse retângulo costuma ser descrito como (x, y, w, h), onde o ponto $(x, y)^T$ é o vértice superior esquerdo do retângulo, w é a largura e h é a altura.

Os algoritmos clássicos para realizar a detecção de pedestres costumavam utilizar *Histograma de Gradientes Orientados* - Histogram of Oriented Gradients *(HOG)* em conjunto com o algoritmo de aprendizagem *Máquina de Vetores de Suporte* - Support Vector Machine *(SVM)*. Os HOGs descrevem os elementos da imagem ao dar ênfase à orientação das bordas de cada parte, e são utilizados para treinar a SVM. A Figura 9 ilustra o algoritmo de Dalal e Triggs (2005). Eles extraem o HOG de uma imagem de teste e ponderam pelos pesos obtidos na SVM, que realiza a classificação binária indicando se o objeto é um pedestre ou não. Em cenas com muita oclusão, o HOG poderia falhar por não conseguir detectar partes importantes das pessoas.

Figura 9 – Detecção de pedestres utilizando HOG de Dalal e Triggs (2005). Da esquerda para a direita, a primeira imagem mostra o gradiente médio das imagens de treino. A segunda imagem mostra em branco quais locais possuem os pesos positivos na SVM (que são mais relevantes para realizar a classificação), enquanto a terceira imagem enfatiza os pesos negativos. A quarta imagem é uma imagem de teste, e a quinta imagem é o seu HOG computado. A sexta e a sétima imagem são o HOG ponderado pelos pesos positivos e negativos respectivamente.

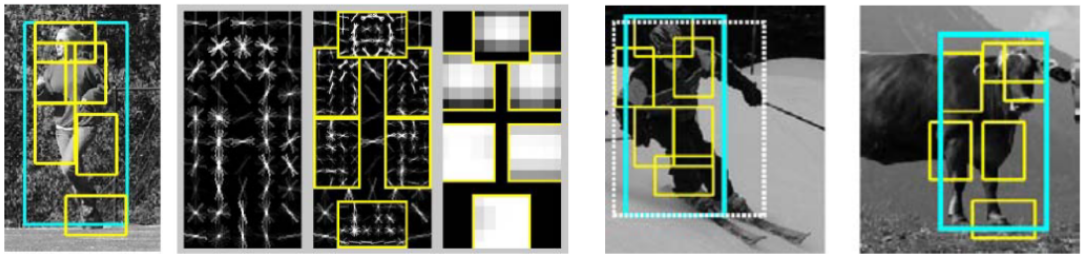


Fonte: DALAL; TRIGGS (2005)

Por conta disso, Felzenszwalb, McAllester e Ramanan (2008) estenderam o método de HOG para realizar a detecção baseado em partes da imagem em vez de considerar o todo. Os modelos são treinados para reconhecer partes dinâmicas dentro da caixa delimitadora do

pedestre. Isso permite que as partes sejam mais ricas em detalhes, aumentando a acurácia do reconhecimento, apesar de ainda ocorrer algumas falhas. A Figura 10 mostra algumas partes detectadas dentro da caixa delimitadora, bem como seus HOGs, e resultados corretos e incorretos de detecção de pedestre.

Figura 10 – Detecção de pedestres utilizando HOG baseado em partes de Felzenszwalb, McAllester e Ramanan (2008). Da esquerda para a direita, um pedestre é detectado (caixa azul) e suas partes são identificadas (caixas amarelas). Cada parte possui um HOG de alta dimensionalidade. Uma pessoa é detectada corretamente, mas uma vaca também é identificada como pedestre incorretamente.



Fonte: FELZENSZWALB; MCALLESTER; RAMANAN (2008)

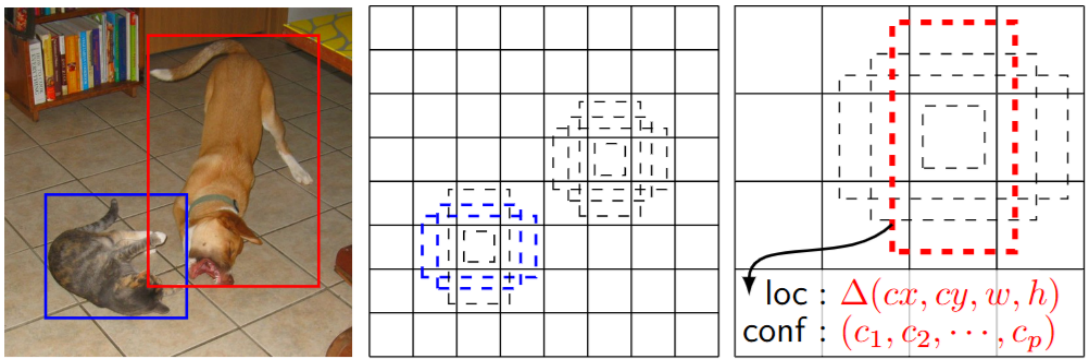
Contudo, os métodos que utilizam HOG são muito sensíveis a variações de tamanho e rotação. A maior parte dos métodos atuais utilizam aprendizagem profunda para conseguir detectar pessoas, pois eles possuem maior precisão e são mais robustos a variações. Eles se apoiam no uso de redes neurais convolucionais, que extraem características da imagem ao passar um filtro por cada região e realizar cálculos. As arquiteturas podem ser divididas em dois tipos, métodos de detecção de um estágio ou de dois estágios.

Quando a detecção é dividida em dois estágios, no primeiro momento o algoritmo apenas detecta possíveis regiões onde há um objeto, e em um segundo momento ele classifica e refina a caixa delimitadora. Por outro lado, quando há apenas um estágio, o algoritmo retorna tanto a localização quanto a classificação do objeto de uma vez só. Por conta disso, a detecção de um estágio é mais rápida, mas tem menor acurácia. Enquanto isso, os métodos de dois estágios enfatizam a precisão em detrimento do desempenho.

Os métodos populares de um estágio são *Detector Multicaixa de Disparo Único* - Single-shot Multibox Detector *(SSD)* (LIU et al., 2016) e *You Only Look Once (YOLO)* (REDMON et al., 2016). A técnica SSD define previamente um conjunto inicial de caixas delimitadoras para utilizar como candidatas, e para cada uma calcula a sua classe e a sua confiança. Eles associam as predições com as anotações verdadeiras considerando que a sua sobreposição, *Interseção sobre União* - Intersection Over Union *(IOU)*, é superior a 0,5, e calculam o erro de confiança em relação às classes e o erro de localização do centro da caixa delimitadora. A

Figura 11 ilustra o método.

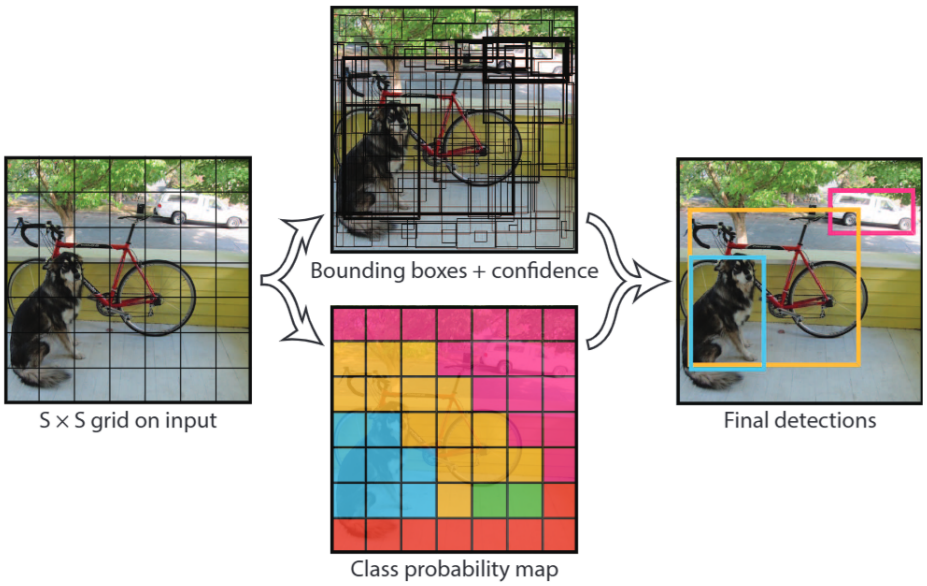
Figura 11 – Detecção de objetos de Liu et al. (2016). A primeira imagem contém a anotação verdadeira de duas caixas delimitadoras envolvendo animais. O método SSD considera mapas de diferentes escalas (nesse caso, um de 8x8 na segunda imagem e um de 4x4 na terceira imagem), e em cada localização dentro do mapa considera as caixas delimitadoras pré-definidas (quatro nesse exemplo) e tenta inferir a localização e a confiança para cada classe.



Fonte: LIU et al. (2016)

Enquanto isso, o YOLO divide a imagem em uma grade e infere a probabilidade de classe para cada célula. Como é possível ter mais de um objeto em uma célula, de forma similar ao SSD, eles utilizam caixas delimitadoras pré-definidas para estimar a localização exata de cada objeto. A Figura 12 demonstra a inferência utilizando YOLO.

Figura 12 – Detecção de objetos utilizando YOLO (REDMON et al., 2016). Cada célula ajuda a inferir caixas delimitadoras de diferentes tamanhos junto com um nível de confiança.



Fonte: REDMON et al. (2016)

2.3 RASTREAMENTO DE PEDESTRES

O rastreamento de pedestres possui um objetivo mais abrangente do que a detecção. Além de localizar as pessoas, é necessário re-localizar as mesmas pessoas individualmente ao longo do tempo. Ou seja, cada indivíduo possui uma identidade única id, que pode ser por exemplo um número, e sempre que ele aparecer devemos determinar a sua posição e a sua identidade corretamente.

Como brevemente mencionado no Capítulo 1, os algoritmos de rastreamento costumam utilizar a estrutura TBD, onde se apoiam em detectores de objetos. A cada instante de tempo t, um algoritmo de detecção estima as possíveis caixas delimitadoras (x, y, w, h) de pedestres. Em seguida, o rastreador agrupa as detecções de quadros diferentes que possuem uma probabilidade maior de pertencer à mesma pessoa, e atribui o mesmo id. Dessa forma, o resultado do rastreamento é um conjunto de detecções D, onde cada detecção i é definida por $d_i = (t_i, id_i, x_i, y_i, w_i, h_i)$. Ao selecionar as detecções que possuem o mesmo id, conseguimos visualizar a trajetória do pedestre ao longo do tempo. A Figura 13 mostra um exemplo de visualização da trajetória dos pedestres.

Figura 13 – Exemplo de visualização das trajetórias dos pedestres obtidas após o rastreamento.

Fonte: NEGRI; GARAYALDE (2017)

Para associar as detecções à mesma trajetória, os algoritmos costumam explorar as características espaciais e visuais das detecções. Do ponto de vista espacial, a localização de

um pedestre entre um instante de tempo e outro não deve ser muito distante. Portanto, as detecções mais próximas possuem uma probabilidade maior de pertenceram à mesma pessoa. Além disso, alguns métodos estabelecem uma distância máxima entre um quadro e outro para permitir a associação da detecção, que é definida por

$$d_{max} = \frac{max\ speed}{FPS}, \tag{2.9}$$

onde $max\ speed$ é a velocidade máxima de uma pessoa em metros por segundo e FPS é a frequência de quadros por segundo do vídeo.

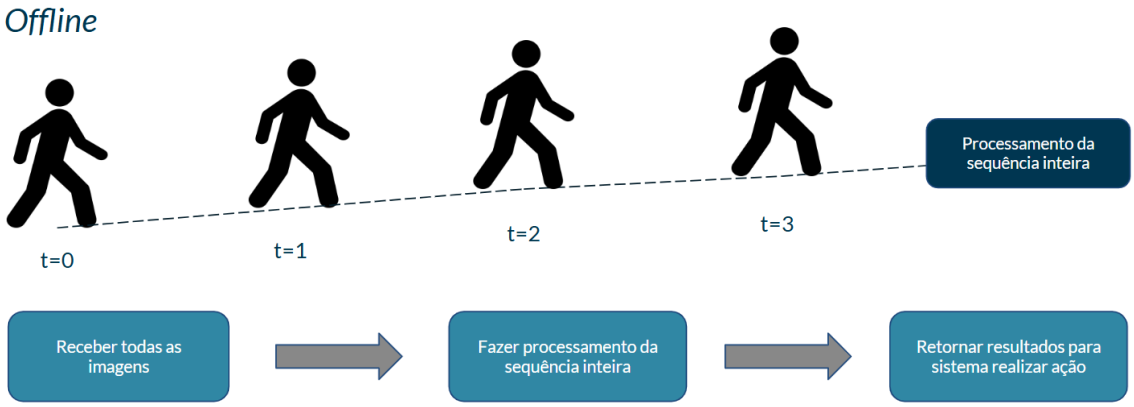
Já as características visuais dos pedestres podem ser obtidas por meio de arquiteturas de aprendizagem profunda baseadas em redes neurais convolucionais. Nessas arquiteturas, os pixels das imagens são manipulados de forma a obter um vetor numérico que represente a aparência do pedestre. O princípio é que ao extrair as características de duas imagens de detecções com aparência similar, os vetores também serão similares. Assim, a detecção passa a ter um atributo a mais para descrevê-la, sendo escrita como $\mathbf{d}_i = (t_i, id_i, x_i, y_i, w_i, h_i, \mathbf{v}_i^n)$, onde $\mathbf{v}_i^n$ é um vetor de dimensão n. Considerando uma detecção $\mathbf{d}_i$ e uma detecção $\mathbf{d}_j$, a similaridade entre os vetores $\mathbf{v}_i^n$ e $\mathbf{v}_j^n$ pode ser calculada através da similaridade por cosseno, determinada por

$$\mathrm{sim}_{\cos}(\mathbf{v}_i^n, \mathbf{v}_j^n) = \frac{\mathbf{v}_i^n \cdot \mathbf{v}_j^n}{\|\mathbf{v}_i^n\|\|\mathbf{v}_j^n\|}. \tag{2.10}$$

Ademais, os algoritmos de rastreamento podem atuar de forma *offline* ou *online*. Os algoritmos *offline* apenas realizam a associação após obterem as detecções de todos os quadros. Ou seja, não é possível obter as identidades dos pedestres simultaneamente à chegada das imagens, sendo necessário esperar que a sequência temporal a ser analisada esteja completa. O processamento *offline* atua de forma linear, e a Figura 14 ilustra ele. Como o rastreamento pode ser visto como um problema de otimização combinatória, onde deve-se encontrar a associação global ótima das trajetórias, os algoritmos *offline* possuem vantagem em relação à acurácia, pois eles tem acesso às informações completas para calcular a associação ótima.

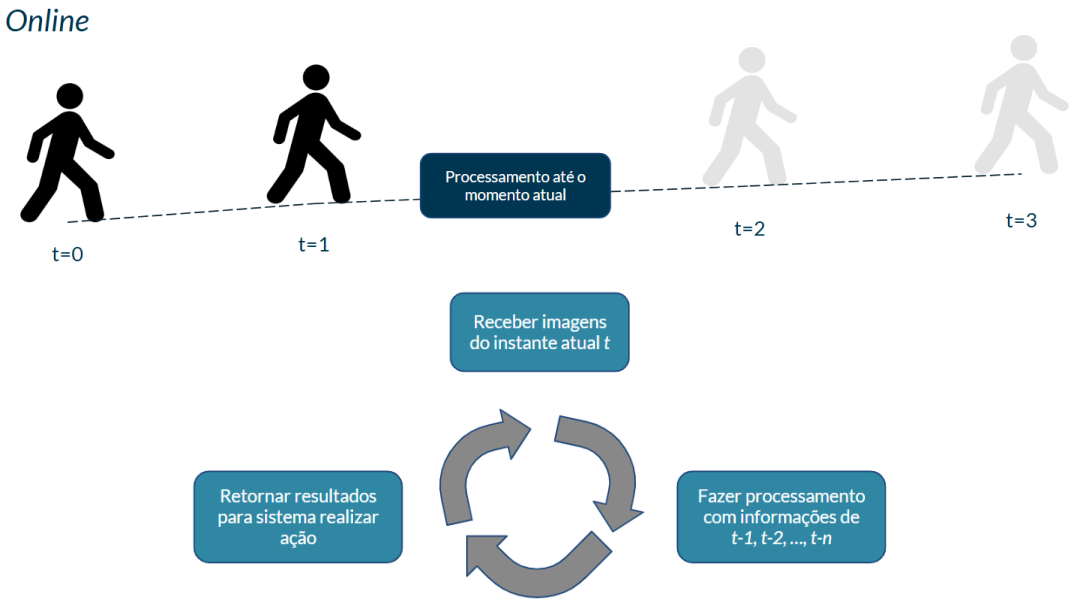
Os algoritmos *online* analisam as detecções à medida que chegam. No primeiro instante de tempo t_0, as trajetórias são inicializadas com uma identidade, e nos instantes seguintes o algoritmo analisa cada detecção nova para determinar se ela pertence ao mesmo pedestre já inicializado. Assim, o algoritmo atua de forma iterativa, como mostra a Figura 15. A cada instante de tempo t, ele extrai as detecções do tempo atual e tenta associar às trajetórias presentes nos quadros anteriores $\{t-1, t-2, ..., t-n\}$.

Figura 14 – Processamento do rastreamento *offline*. As etapas devem ser seguidas de forma linear, onde primeiro é necessário receber todas as imagens da sequência, depois o rastreamento é feito sobre toda a sequência e por fim os resultados são obtidos.



Fonte: Elaborada pela autora (2024)

Figura 15 – Processamento do rastreamento *online*. As etapas ocorrem de forma iterativa, onde a cada instante t o rastreamento é feito considerando as detecções que temos até o momento e já é possível utilizar os resultados.



Fonte: Elaborada pela autora (2024)

3 TRABALHOS RELACIONADOS

Este capítulo descreve os trabalhos relacionados que realizam o rastreamento de pedestres utilizando apenas uma câmera ou utilizando múltiplas câmeras.

3.1 RASTREAMENTO COM CÂMERA ÚNICA

A área de rastreamento de câmera única vem sendo explorada em diversos aspectos. Yi et al. (2024) buscam resolver o problema de rastrear em câmeras com movimento e propõem um rastreador que aplica os mesmos parâmetros de compensação de movimento da câmera ao longo de toda a sequência de vídeo, utilizando um filtro de Kalman no plano do solo. Além disso, eles propõem uma nova medida de distância baseada na distância de Mahalanobis, o que também melhora a associação de dados em cenários com movimento da câmera.

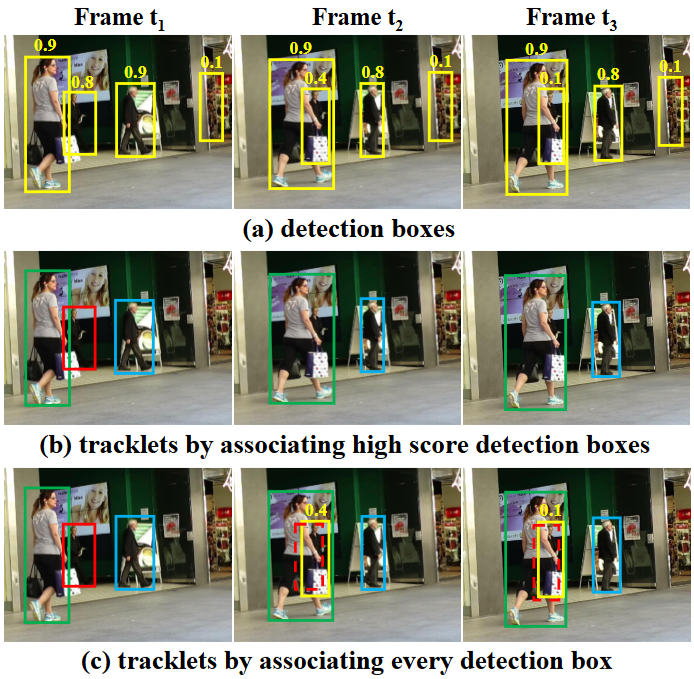
Wang et al. (2024) desenvolveu um módulo de aprendizado de similaridade baseado em rede siamesa, no qual a rede neural recebe duas detecções e estima a sua similaridade, em oposição a métodos que devolvem um vetor descritivo de aparência.

Segundo Cao et al. (2023), métodos baseados em filtro de Kalman assumem que os objetos se movem linearmente, o que pode ser impreciso durante períodos prolongados de oclusão. Com isso, eles investigam um método que utiliza observações dos objetos (medidas pelo detector de objetos) para calcular uma trajetória virtual durante o período de oclusão, corrigindo a acumulação de erros dos parâmetros do filtro.

Para melhorar a acurácia do rastreamento, alguns métodos de rastreamento de câmera única propuseram formas para lidar com detecções faltantes. Normalmente, as detecções são acompanhadas por uma taxa de confiança, e apenas as que possuem a confiança alta são utilizadas. Utilizar detecções com um baixo nível de confiança pode diminuir o número de detecções faltantes, porém, também aumenta as chances de indicar uma detecção onde não há ninguém, levando a um falso positivo.

A técnica ByteTrack (ZHANG et al., 2022) inclui detecções com baixa confiança e tenta decidir quais são verdadeiras durante o rastreamento. Eles realizam a associação de detecções em duas etapas. Na primeira etapa, apenas as detecções com maior confiança são consideradas. Já na segunda etapa, eles tentam associar as detecções de baixa confiança às trajetórias restantes.

Figura 16 – Demonstração do ByteTrack (ZHANG et al., 2022). O rastreador recebe as detecções e suas taxas de confiança. Em seguida, associa primeiro as detecções com alta taxa de confiança, e depois associa as restantes.

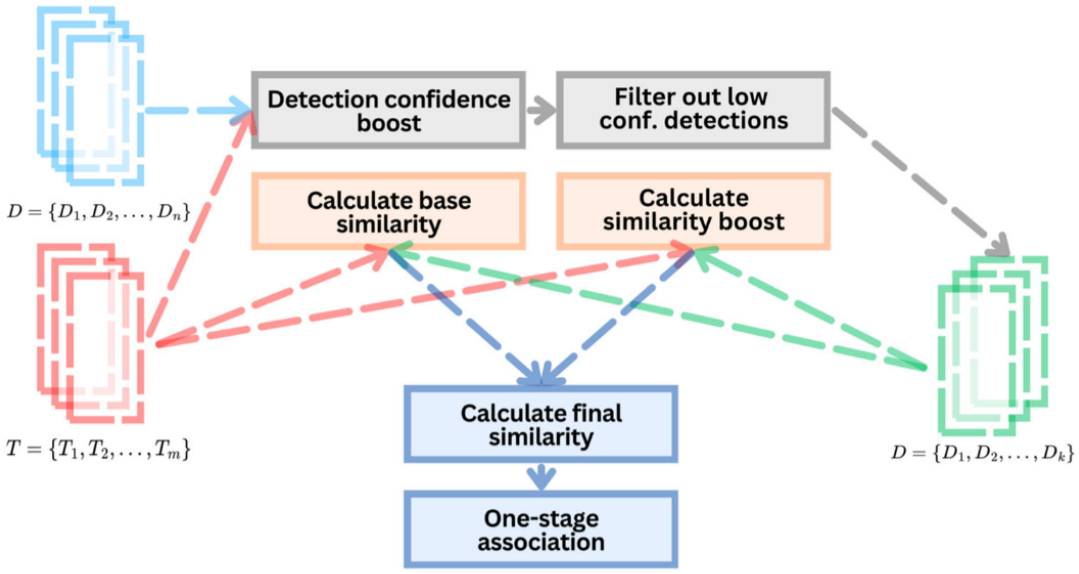


Fonte: ZHANG et al. (2022)

O método BoostTrack (STANOJEVIC; TODOROVIC, 2024), além de utilizar a taxa de confiança das detecções, propõe uma taxa de confiança para as trajetórias, considerando as trajetórias que tiveram uma atualização mais recente como mais confiáveis. A confiança de trajetória é multiplicada pela confiança de detecção e somada à matriz de similaridade entre as novas detecções e as trajetórias existentes, de forma a considerar que detecções-trajetórias com maior confiança devem ter prioridade na associação de identidade. Além disso, eles aumentam a confiança de detecções que tinham um baixo nível de confiança, por exemplo por estarem parcialmente oclusas, mas tinham a posição similar com a posição prevista pelo filtro de Kalman (KALMAN, 1960). A Figura 17 ilustra o BoostTrack.

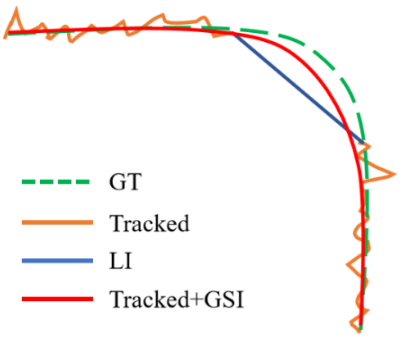
Enquanto isso, o método StrongSORT (DU et al., 2023) propôs algumas melhorias ao rastreador clássico DeepSORT (WOJKE; BEWLEY; PAULUS, 2017), incluindo um módulo que recupera trajetórias parcialmente perdidas chamado *Interpolação com Suavização Gaussiana - Gaussian-smoothed Interpolation (GSI)*. Se uma pessoa não apareceu no instante t, mas estava presente nos instantes $t-1$ e $t+1$, eles interpolam essas duas detecções para criar uma detecção em t. Em seguida, eles treinam um modelo de *Regressão de Processo Gaussiano - Gaussian Process Regression (GPR)* utilizando as observações anteriores da trajetória para suavizar as coordenadas.

Figura 17 – O método BoostTrack (STANOJEVIC; TODOROVIC, 2024) propõe que a taxa de confiança de detecções que tem a posição similar à posição prevista pelo filtro de Kalman seja aumentada. Além disso, uma taxa de confiança que relaciona detecções e trajetórias é proposta para ser utilizada em conjunto com matrizes de similaridade base que já foram propostas em outros trabalhos.



Fonte: STANOJEVIC; TODOROVIC (2024)

Figura 18 – Comparação da trajetória dos pedestres nas anotações verdadeiras (verde) com a trajetória dos pedestres usando diferentes módulos do rastreamento. Inicialmente, o rastreamento ocorre com bastante ruído e mudanças bruscas (laranja), além de deixar de ocorrer durante oclusões. A interpolação linear (azul) recupera o pedestre durante a oclusão. E incluindo a função gaussiana (vermelho) a trajetória rastreada se torna mais suave e próxima das anotações verdadeiras.



Fonte: DU et al. (2023)

Como esse processo utiliza informações de quadros futuros, ele é *offline*. Além disso, tanto o ByteTrack quanto o StrongSORT foram aplicados apenas no cenário de câmera única, dependendo de um único ponto de vista.

Neste trabalho, nós melhoramos o módulo GSI do StrongSORT, adaptando-o para se tornar *online* e multi-câmera.

Além disso, diferentemente do ByteTrack, as detecções extras não são detecções de baixa confiança advindas do detector. São extrapolações de detecções de alta confiança existentes

nos quadros anteriores.

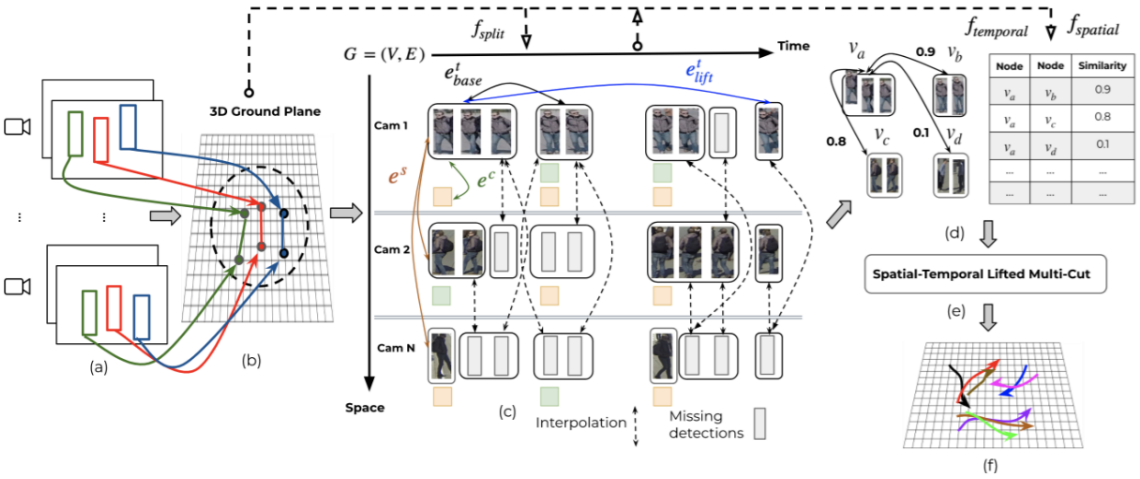
3.2 RASTREAMENTO COM MÚLTIPLAS CÂMERAS

Alguns métodos utilizam múltiplas câmeras para rastrear pedestres de forma *offline*. Nguyen et al. (2022) projetam as detecções de cada câmera no plano de chão utilizando a calibração das câmeras. Em seguida, criam um grafo onde os nós são trajetórias da mesma pessoa na mesma câmera ao longo do tempo. Eles obtém essas trajetórias utilizando rastreadores de câmera única. Depois, eles ainda refinam os resultados utilizando uma rede neural que infere a probabilidade das detecções de instantes consecutivos pertencerem à mesma pessoa. Caso a probabilidade entre as detecções de $t-1$ e t seja baixa, eles particionam a trajetória em duas, $\{t-n, ..., t-1\}$ e $\{t, ..., t+m\}$. Então, eles calculam as afinidades temporais e espaciais e utilizam como entrada de uma rede neural temporal e uma espacial que resultam em uma pontuação de similaridade entre cada nó. As afinidades incluem uma afinidade entre a última/primeira detecção da trajetória e a extrapolação da trajetória para o quadro seguinte ou para o quadro anterior, uma afinidade entre a aparência das detecções utilizando a similaridade por cosseno entre os vetores descritores de aparência, e uma afinidade com a distância das detecções no plano de chão 3D. Após obter as pontuações de similaridade temporal e espacial, elas são utilizadas como peso das arestas do grafo, e um algoritmo de acoplamento é utilizado para associar a maior quantidade de detecções com o menor custo. A Figura 19 ilustra esse método.

Enquanto isso, Vo et al. (2021) foca em melhorar descritores da aparência dos pedestres ao realizar *fine-tuning* dos modelos em vídeos sem anotações utilizando um método auto-supervisionado. A Figura 20 demonstra os resultados do descritor de aparência. A aparência dos pedestres é utilizada durante o rastreamento para agrupar detecções similares utilizando um grafo do vizinho mais próximo *(k-NN graph)*.

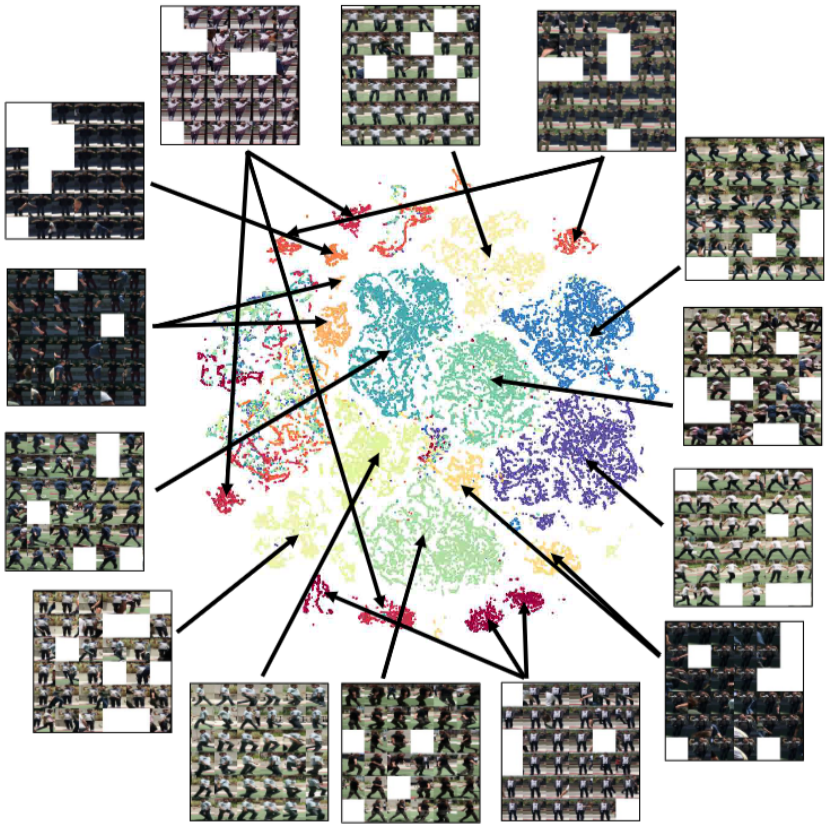
Por outro lado, You e Jiang (2020) propôs um processo *online* fim-a-fim que realiza tanto a detecção quanto o rastreamento. Primeiro, eles treinam uma rede neural para estimar o mapa de calor com o ponto de chão dos pedestres em cada câmera. Em seguida, eles utilizam a calibração das câmeras para projetar o mapa de calor no plano de chão, e eles são agregados fazendo a média ou empilhando. Esse processo gera muitos falsos positivos, então eles utilizam ainda um modelo de aprendizagem profunda que classifica se as detecções são positivas ou não. Em seguida, eles realizam o rastreamento de forma eficiente ao criar um grafo onde as

Figura 19 – Algoritmo de Nguyen et al. (2022). (a) As caixas delimitadoras são entrada para o método. (b) As coordenadas são projetadas no plano de chão 3D. (c) Um grafo espaço-temporal é criado com as detecções. (d) Os custos de associação entre os nós são computados utilizando redes neurais de grafos. (e) A associação global de trajetórias é feita. (f) As coordenadas 3D das trajetórias são geradas.



Fonte: NGUYEN et al. (2022)

Figura 20 – Demonstração de de Vo et al. (2021). As detecções de aparência similar são agrupadas.

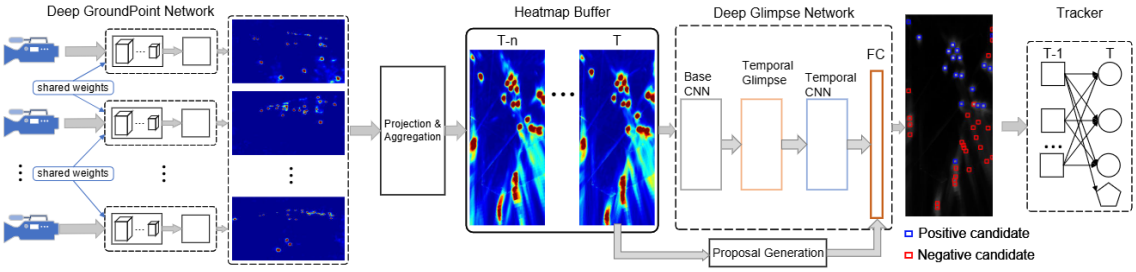


Fonte: VO et al. (2021)

detecções do instante $t-1$ estão conectadas às detecções do instante t por uma aresta com peso igual à distância Euclidiana entre as respectivas localizações. As detecções são acopladas

ao utilizar um algoritmo que associe a maior quantidade de nós com o menor custo possível, nesse caso considerando a menor distância. A Figura 21 ilustra o método.

Figura 21 – Algoritmo de You e Jiang (2020). Uma rede neural estima o mapa de calor com as detecções em cada câmera. Em seguida, eles são projetados para o plano de chão. Eles utilizam uma rede neural para classificar se as detecções são positivas para remover falsos positivos, e realizam o rastreamento criando um grafo bipartido e fazendo o acoplamento.



Fonte: YOU; JIANG (2020)

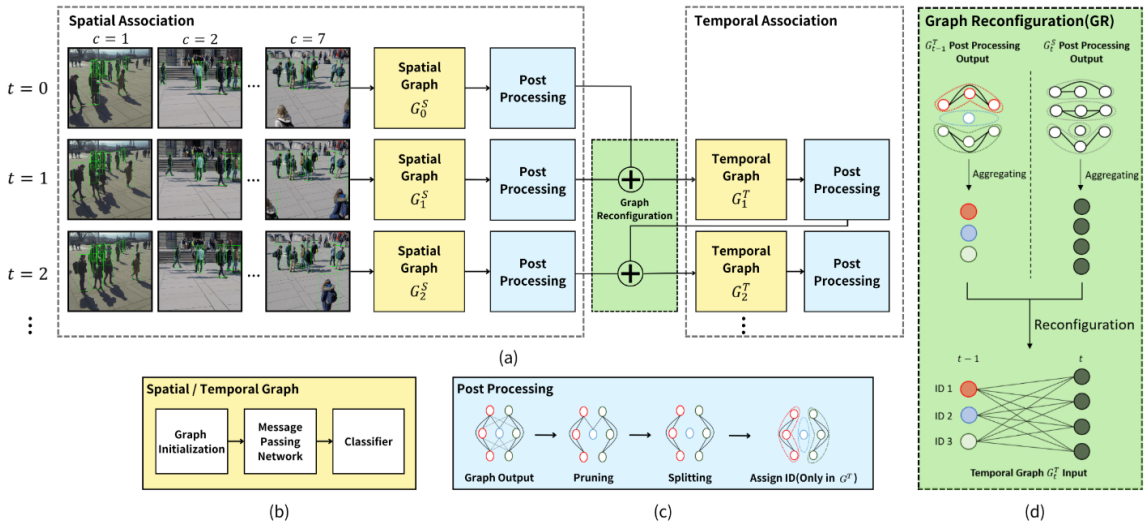
Cheng et al. (2023) também propôs um rastreador *online* (Figura 22). Ele utiliza redes neurais de grafos para associar as detecções tanto entre as câmeras quanto entre os quadros. Eles utilizam como atributos das arestas a distância entre a posição das detecções, entre os vetores descritores de aparência, e entre a velocidade. Além disso, eles treinam uma rede neural de grafo para associar as detecções espacialmente (entre câmeras) e outra para associar temporalmente. As redes são treinadas para inferir se a aresta entre duas detecções é positiva (pertencem à mesma pessoa) ou negativa. Após obter as associações temporais e espaciais, eles agregam o grafo temporal do instante anterior G^T_{t-1} e o grafo espacial do instante atual G^S_t para formar o grafo temporal do instante atual G^T_t, transformando os nós já conectados em apenas um nó, onde seus atributos são a média dos antigos nós, e então G^T_t é um grafo bipartido entre os vértices de $t-1$ vindos do grafo temporal e t vindos do grafo espacial.

Andrade e Lima (2023) propôs um rastreador que realiza apenas a associação temporal por redes neurais de grafos. Contudo, também é demonstrado que a rede pode ser treinada com pseudo-rótulos gerados por um rastreador não supervisionado. O método é ilustrado na Figura 23.

Contudo, os métodos mencionados anteriormente utilizam redes neurais que precisam ser treinadas. Para treiná-las, normalmente é necessário coletar anotações verdadeiras com as posições dos pedestres, o que é um processo trabalhoso. Mesmo se um método for auto-supervisionado e não precisar de anotações, o treinamento de uma rede neural atrasa a aplicação do rastreamento em novos ambientes.

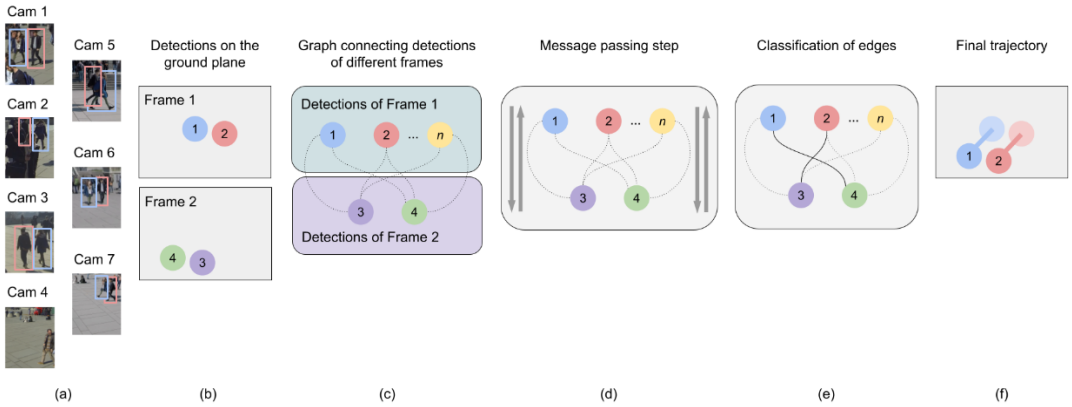
Ong et al. (2022) desenvolveu um rastreador que não requer treinamento e possui um

Figura 22 – Algoritmo de Cheng et al. (2023). (a) Associação espaço-temporal de detecções utilizando redes neurais de grafos. (b) Arquitetura da rede neural de grafo. (c) Pós-processamento de trajetórias. (d) Agregamento de grafos temporais e espaciais.



Fonte: CHENG et al. (2023)

Figura 23 – Algoritmo de Andrade e Lima (2023). (a) As caixas delimitadoras dos pedestres são entrada para o algoritmo. (b) Um detector 3D que devolve as detecções no plano de chão é utilizado. (c) Um grafo é criado conectando detecções de quadros diferentes. (d) A rede é treinada ao compartilhar informações entre nós vizinhos e arestas. (e) As arestas são classificadas como ativas ou inativas. (f) O algoritmo devolve as trajetórias no plano de chão.

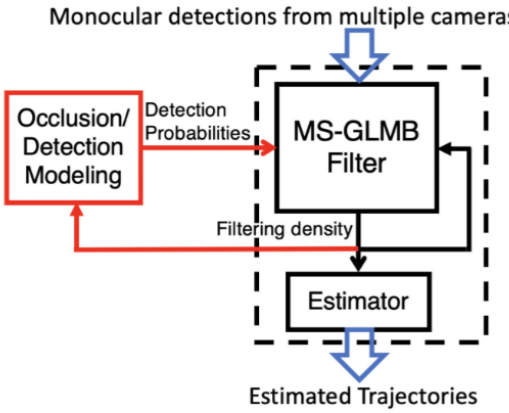


Fonte: ANDRADE; LIMA (2023)

modelo para lidar com oclusões que integra de forma eficiente várias tarefas, como gerenciamento de trajetórias, estimativa de estado, e tratamento de oclusões e falhas de detecção. A Figura 24 apresenta o método.

Lyra. et al. (2022) propôs um rastreador *online* não supervisionado, representado na Figura 25. Ele utiliza a distância entre as detecções recebidas para calcular quais deveriam ser vinculadas. Se o pedestre não for detectado em um quadro, o rastreador simplesmente não inclui a sua posição. Como o rastreador possui acesso a informações temporais, seria interes-

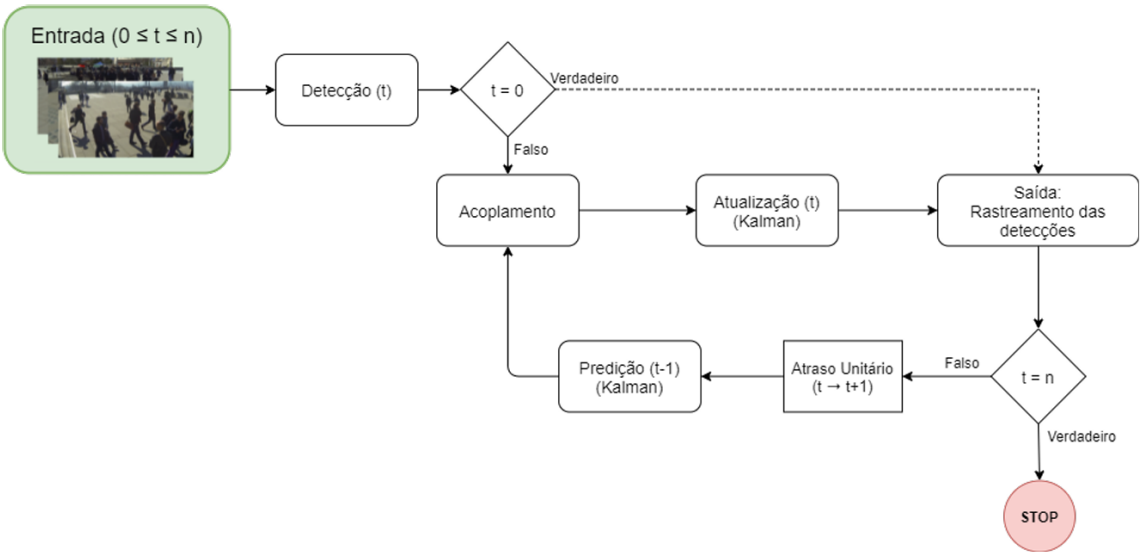
Figura 24 – Processamento de Ong et al. (2022). O filtro de Bayes estima as trajetórias e um módulo auxiliar gerencia as oclusões.



Fonte: ONG et al. (2022)

sante que o algoritmo de rastreamento tentasse auxiliar o trabalho do detector e amenizar erros como detecções faltantes.

Figura 25 – Algoritmo de Lyra. et al. (2022). Um grafo é criado com as detecções, e o algoritmo de acoplamento associa as detecções mais perto. A posição das detecções é atualizada com o filtro de Kalman.



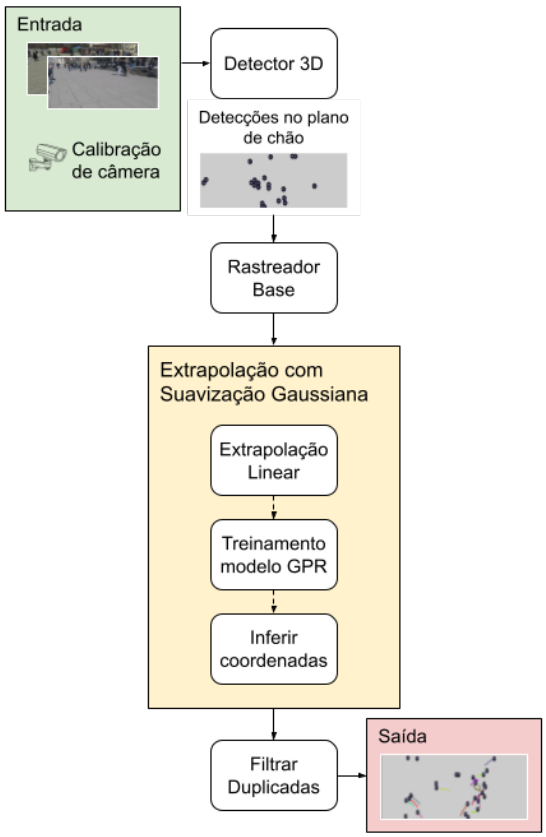
Fonte: LYRA. et al. (2022)

Portanto, propomos um módulo que pode ser acoplado a rastreadores multi-câmera sem precisar de treinamento. Nosso módulo recupera pedestres que passaram despercebidos pelo detector, combinando extrapolação linear com GPR. Além disso, o módulo é *online* para ser adequado a aplicações de tempo real.

4 RETRYTRACK

Este capítulo explica nosso método de recuperação de pedestres e o procedimento de rastreamento completo, que contém quatro etapas e é ilustrado na Figura 26. Primeiro, extraímos detecções das imagens (Seção 4.1). Depois, utilizamos as detecções como entrada para o rastreador base (Seção 4.2). Finalmente, introduzimos um módulo para recuperar detecções perdidas (Seção 4.3) e usamos um filtro para remover duplicatas (Seção 4.4).

Figura 26 – Imagens de múltiplas câmeras são entrada para o detector, que devolve as detecções como coordenadas de mundo no plano de chão. Um rastreador base calcula as trajetórias dos pedestres, e nossa extrapolação gaussiana suavizada refina os resultados, recuperando detecções ausentes. A saída são as trajetórias de pedestres atualizadas.



Fonte: Elaborada pela autora (2024)

4.1 DETECTOR DE PEDESTRES MULTI-CÂMERA

Usamos o rastreador proposto por Lyra. et al. (2022) como base para nosso método. Ele se baseia em uma detecção de pedestres multi-câmera proposta por Lima et al. (2021).

Neste detector, as caixas delimitadoras dos pedestres são detectadas pelo modelo de apren-

dizagem profunda YOLOv3 (REDMON; FARHADI, 2018). Em seguida, a biblioteca AlphaPose (LI et al., 2019) extrai 17 pontos-chave do corpo do humano, e os dois pontos dos tornozelos são usados para calcular o ponto de chão do pedestre na câmera.

Como o tornozelo fica um pouco acima do ponto de chão, é necessário deslocar os pontos para baixo. O montante que deve ser deslocado é calculado por

$$\delta = bb_{y_{max}} - \max(kl_y, kr_y), \tag{4.1}$$

onde $bb_{y_{max}}$ é o ponto máximo do eixo y da caixa delimitadora, e kl_y e kr_y são o componente y dos pontos dos tornozelos esquerdo e direito respectivamente.

O ponto de chão do pedestre na câmera $\mathbf{p}_{cam} = (x_{cam}, y_{cam})^T$ é o ponto médio entre o ponto do tornozelo esquerdo e o ponto do tornozelo direito deslocados δ para baixo. Ele é obtido por

$$\mathbf{p}_{cam} = \left(\left(\frac{kl_x + kr_x}{2} \right), \left(\frac{(kl_y + \delta) + (kr_y + \delta)}{2} \right) \right). \tag{4.2}$$

Em seguida, eles utilizam os parâmetros $\mathbf{K}$ e $[\mathbf{R}|\mathbf{t}]$ da câmera para converter cada coordenada de câmera $\mathbf{p}_{cam}$ em coordenadas 3D mundiais no plano de chão $\mathbf{p}_{world} = (X, Y, 0)^T$. Assumindo que o chão corresponde a $Z = 0$, a projeção é dada por

$$\begin{pmatrix} x_{cam} \\ y_{cam} \\ 1 \end{pmatrix} = \mathbf{K}[\mathbf{R}^1\mathbf{R}^2\mathbf{R}^3\mathbf{t}] \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} \tag{4.3}$$

$$= \mathbf{K}[\mathbf{R}^1\mathbf{R}^2\mathbf{t}] \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{4.4}$$

$$= \mathbf{H}^{-1} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}, \tag{4.5}$$

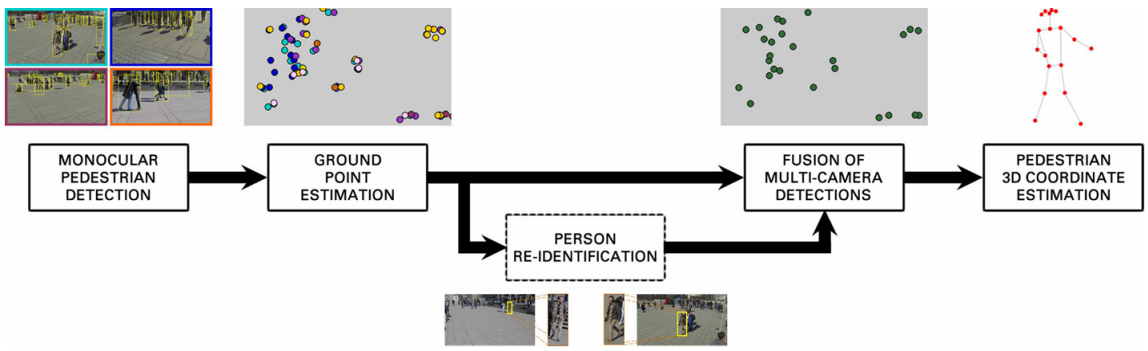
onde $\mathbf{R}^i$ é a i-ésima coluna de $\mathbf{R}$ e $\mathbf{H}$ é a matriz de homografia que mapeia as coordenadas de imagem para as coordenadas de mundo no chão.

Cada câmera gerou um $\mathbf{p}_{world}$ por pedestre. Para fundir os pontos, eles criam um grafo G onde os vértices são $\mathbf{p}_{world}$ de diferentes câmeras e as arestas conectam pontos cuja distância

euclidiana é menor que um limiar. Opcionalmente, um vetor de re-identificação é gerado através de redes neurais convolucionais (LECUN et al., 1998) para representar as características visuais da detecção, e as arestas também representam a similaridade por cosseno entre os vetores das detecções de suas pontas. O grafo é repartido em subconjuntos se baseando no problema da cobertura de clique (KARP, 1972). Um clique é um subconjunto de vértices onde cada par de vértices distintos está conectado. O problema da cobertura de clique busca particionar os vértices do grafo no menor número de cliques. Nesse trabalho, um clique representa um pedestre. Para resolver o problema de clique em tempo linear, eles veem o problema como a coloração do grafo complementar de G e utilizam um algoritmo guloso para colorir (KUBALE, 2004).

Assim, após conectar as detecções de câmeras diferentes que pertencem ao mesmo pedestre resolvendo o problema da cobertura de clique, eles estimam a coordenada final do pedestre no espaço 3D $\mathbf{p}_{ped} = (x_{world}, y_{world}, z_{world})$ ao escolher os pontos 3D de vistas pares que minimizam o erro de re-projeção em todas as vistas, e calculando a projeção ortogonal da média dessas coordenadas. Todos os pedestres estão no chão, que está localizado em $z_{world} = 0$ (LIMA et al., 2021). O método de detecção é ilustrado na Figura 27.

Figura 27 – Processo de detecção. Os pedestres são detectados em cada câmera, e em seguida o ponto no plano de chão é estimado a partir de cada câmera. Os pontos de chão de diferentes câmeras são agrupados considerando a distância euclidiana, e opcionalmente a distância entre os vetores de re-identificação. No final, obtemos a coordenada 3D do pedestre.



Fonte: LIMA et al. (2022)

4.2 RASTREADOR DE PEDESTRES MULTI-CÂMERA BASE

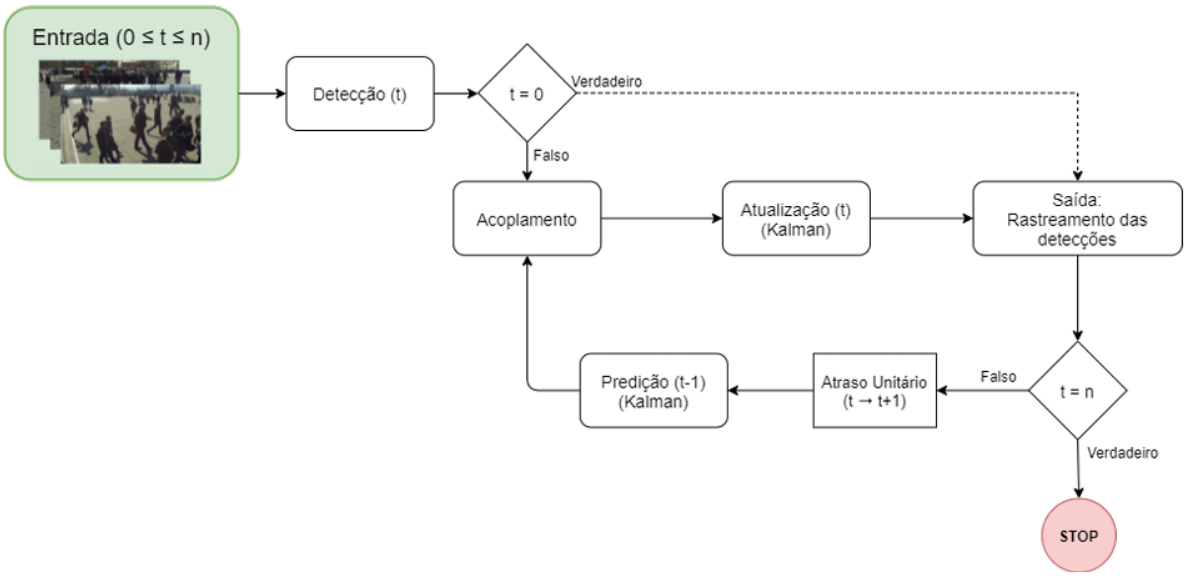
Após obter as detecções, elas são utilizadas como entrada para o rastreador. No primeiro instante de tempo ($t = 0$), cada detecção recebe um id sequencial. Nos quadros seguintes, eles comparam as detecções atuais com as trajetórias existentes.

Um grafo bipartido é construído, onde um conjunto N^A de nós representa as detecções do quadro anterior, um conjunto N^B de nós representa as detecções do quadro atual, e as arestas conectam os nós de N^A com os de N^B. As arestas possuem a distância euclidiana entre elas, e opcionalmente a distância do cosseno entre vetores de aparência.

Se as detecções do quadro anterior não incluírem uma trajetória mais antiga, um filtro de Kalman (KALMAN, 1960) também é utilizado, para prever qual seria a posição da detecção mais recente nesta trajetória.

Após obter o grafo, eles usam o algoritmo de acoplamento de peso máximo (GALIL, 1986) para recuperar quais detecções devem ser atribuídas à mesma pessoa. Caso uma detecção nova não seja associada a nenhuma das trajetórias existentes, um novo id é atribuído. O algoritmo é ilustrado na Figura 28.

Figura 28 – Processo de rastreamento. A cada instante de tempo, o algoritmo recebe as imagens das múltiplas câmeras e utiliza o detector mencionado na Seção 4.1. Se estiver no primeiro instante de tempo, atribui identidades iniciais. Se não, monta o grafo e utiliza o algoritmo de acoplamento. Em seguida, atualiza o estado do objeto no filtro de Kalman, e retorna o resultado do rastreamento. Antes de realizar o próximo acoplamento, o filtro de Kalman é utilizado para prever a posição de pedestres e obter uma localização mais aproximada de onde eles estariam no quadro atual utilizando sua velocidade anteriormente. O processo se repete até chegar ao instante final da sequência.



Fonte: LYRA. et al. (2022)

4.3 EXTRAPOLAÇÃO COM SUAVIZAÇÃO GAUSSIANA

Ao final de cada quadro, aplicamos o módulo de Extrapolação com Suavização Gaussiana - *Gaussian-smoothed Extrapolation* (GSE) para recuperar rastros perdidos. As etapas do módulo

estão ilustradas na Figura 29. Se uma pessoa do quadro anterior $t-1$ não for encontrada no tempo atual t, usamos extrapolação linear (BREZINSKI; ZAGLIA, 2013) com as detecções de $t-2$ e $t-1$ para gerar uma nova detecção no tempo t usando a seguinte equação

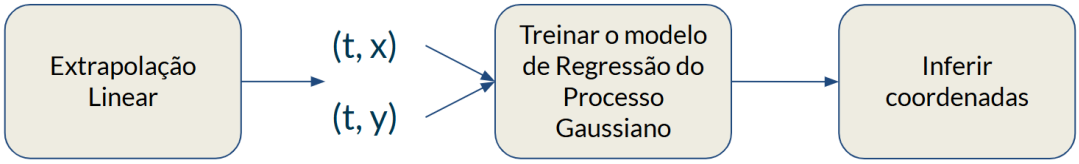
$$(x_t, y_t) = (x_{t-2}, y_{t-2}) + f * ((x_{t-1}, y_{t-1}) - (x_{t-2}, y_{t-2})), \tag{4.6}$$

onde

$$f = \frac{f_t - f_{t-2}}{f_{t-1} - f_{t-2}}, \tag{4.7}$$

f_t é o número do quadro t, e (x, y) é a coordenada de mundo.

Figura 29 – Ilustração das etapas do módulo GSE. Primeiro ocorre a extrapolação linear, que gera uma nova coordenada para pedestres que não foram identificados no tempo atual. Em seguida, as observações de quadros anteriores são usadas para treinar o modelo de GPR. Por fim, inferimos as coordenadas suavizadas.



Fonte: Elaborada pela autora (2024)

Como critério de parada, verificamos se a coordenada extrapolada ainda está dentro da Área de Interesse - *Area of Interest* (AOI), um retângulo no plano de chão de mundo no qual queremos rastrear pedestres, definido por $AOI = (x_{min}, x_{max}, y_{min}, y_{max})$. Não incluímos essa coordenada em nossos resultados se ela estiver fora da AOI. Assim, iremos parar de tentar recuperar esse pedestre.

Além disso, definimos um limite para extrapolar o mesmo pedestre. Se o pedestre já foi extrapolado durante os últimos m quadros, onde m é o limite de quadros consecutivos que permitimos a extrapolação, nós a interrompemos. Isso evita a reprodução infinita de caminhos incorretos.

Além disso, os pontos gerados assumem que o movimento do pedestre é linear, o que geralmente não é o caso. Portanto, usamos a GPR para melhorar nossas previsões.

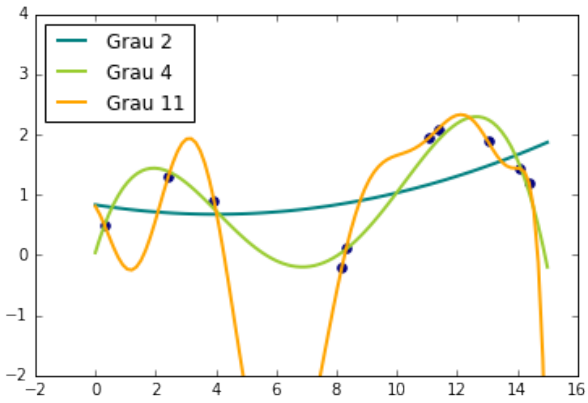
Dado um conjunto de pontos observados, múltiplas funções podem se ajustar a eles, como demonstrado na Figura 30. A GPR é definida como um conjunto de funções de variáveis aleatórias, em que qualquer subconjunto finito de pontos segue uma distribuição gaussiana e calcula a distribuição de probabilidade sobre elas (ZHANG et al., 2023, Capítulo 18). Se traçarmos uma linha reta entre os pontos, o resultado será uma função ruidosa que é inadequada para prever

novos pontos. Portanto, usamos um núcleo de Função de Base Radial - *Radial Basis Function* (RBF) para suavizar as funções. O núcleo pode ser denotado por

$$k(f_i, f_j) = exp\left(-\frac{\|f_i - f_j\|^2}{2\lambda^2}\right), \tag{4.8}$$

onde f é o quadro, e $\lambda = \tau * \log(\tau^3/l)$. l se refere ao comprimento das trajetórias, e τ é definido como 10 (DU et al., 2023). A Figura 31 exemplifica o efeito do núcleo nas funções.

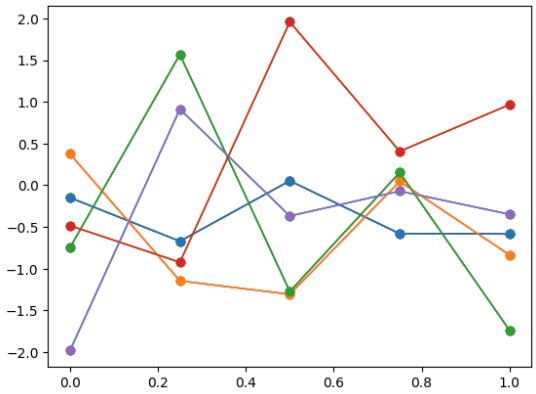
Figura 30 – Imagem ilustrando um conjunto de pontos e múltiplas funções que se ajustam a eles.



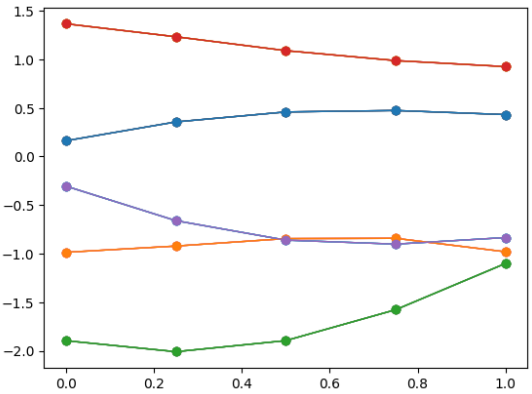
Fonte: Facure (2018)[1]

Figura 31 – Exemplo entre funções antes e depois da aplicação da RBF.

(a) Funções antes de usar RBF.



(b) Funções após usar RBF.



Fonte: Elaborada pela autora (2024)

Depois, a distribuição de probabilidade (WANG, 2023) é calculada usando

$$P(f) = \frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(f-\mu)^2}{2\sigma^2}\right), \tag{4.9}$$

onde f é o quadro, μ é a média e σ^2 é a variância.

[1] <https://lamfo-unb.github.io/2018/03/24/gaussian-process/>. Acessado em 10 de Julho de 2024.

Ajustamos o modelo GPR usando o quadro t como variável e x_{world} e y_{world} como valores alvo. Assim, a função de regressão modelada por GPR é

$$P(X|f) = N(X|\mu, K), \tag{4.10}$$

onde f são os quadros, X são os valores x_{world} e y_{world}, μ é a função média derivada da distribuição de probabilidade e K é a função do núcleo. Assim, prevemos as coordenadas suavizadas.

4.4 FILTRO DE DETECÇÕES DUPLICADAS

Durante os experimentos, descobrimos que algumas coordenadas estavam muito próximas de outra detecção, como na Figura 32. Não precisamos de ocorrências duplicadas dentro de um curto alcance, pois duas pessoas não podem ocupar o mesmo espaço.

Figura 32 – Exemplo de caso onde duas detecções estão muito próximas (#030 e #021).

Fonte: Elaborada pela autora (2024)

Portanto, descartamos detecções que têm menos de 0,4 m de distância, que está dentro da largura típica dos ombros (SHIWAKOTI; SARVI; BURD, 2014).

5 EXPERIMENTOS

Este capítulo explica como o método proposto foi avaliado e quais foram os resultados. A Seção 5.1 descreve a base de dados, e a Seção 5.2, as métricas. Em seguida, a Seção 5.3 compara nossa técnica com o estado da arte. Finalmente, mostramos o estudo de ablação na Seção 5.4.

5.1 BASE DE DADOS

Utilizamos duas bases de dados em nossos experimentos, WILDTRACK (CHAVDAROVA et al., 2018) e MultiviewX (HOU; ZHENG; GOULD, 2020), para validar a nossa solução em diferentes ambientes. Descrevemos nas próximas subseções os seus detalhes e diferenças.

5.1.1 WILDTRACK

A base de dados pública WILDTRACK (CHAVDAROVA et al., 2018) possui sete câmeras com vistas sobrepostas e parâmetros de calibração conhecidos (Figura 33). As câmeras têm 400 quadros com anotações a dois quadros por segundo - *frames per second* (FPS) indicando o ID do pedestre, a caixa delimitadora em cada câmera (se o pedestre estiver visível daquele ponto de vista) e a coordenada 3D do plano de chão de mundo. Cada quadro tem cerca de 20 pedestres.

Figura 33 – Imagens das sete câmeras da base de dados WILDTRACK.

Fonte: CHAVDAROVA et al. (2018)

Como outros métodos precisam dividir quadros para treinamento e para teste, utilizamos apenas os últimos 10% dos quadros para avaliar nossa técnica. Assim, o subconjunto de teste é o mesmo, para termos uma comparação justa.

5.1.2 MultiviewX

O MultiviewX (HOU; ZHENG; GOULD, 2020) é um conjunto de dados sintético, criado por meio do motor gráfico Unity. Seguindo a configuração WILDTRACK, eles anotaram 400 quadros a dois FPS. No entanto, ele tem seis câmeras, e cerca de 40 pedestres por quadro, o que é o dobro do WILDTRACK (Figura 34). Portanto, é um ambiente mais desafiador para o algoritmo. Também usamos os últimos 10% dos quadros em nossa avaliação.

Figura 34 – Imagens das seis câmeras da base de dados MultiviewX.

Fonte: HOU; ZHENG; GOULD (2020)

5.2 MÉTRICAS

Existem três tipos de erros que podem acontecer durante o rastreamento. Primeiro, uma detecção pode apontar para um espaço que não possui pedestre, o que é conhecido como Falso Positivo (FP). Segundo, uma pessoa real pode não ser detectada, o que é chamado de Falso Negativo (FN). Finalmente, se um novo id for atribuído a uma pessoa que já vinha sendo rastreada com outro id, isso é considerado uma Troca de Identidade - *Mismatch* (MM).

Usamos as métricas CLEAR MOT (BERNARDIN; STIEFELHAGEN, 2008) que condensam os erros Acurácia de Rastreamento de Múltiplos Objetos - *Multiple Object Tracking Accuracy* (MOTA) e Precisão de Rastreamento de Múltiplos Objetos - *Multiple Object Tracking Precision* (MOTP). Enquanto o MOTA avalia a proporção entre erros e objetos nas anotações verdadeiras, o MOTP mede a precisão métrica no rastreamento.

O MOTA é calculado por

$$MOTA = 1 - \frac{FP + FN + MM}{OBJ}, \tag{5.1}$$

onde OBJ é a quantidade de objetos que existem nas anotações verdadeiras. De forma simplificada, os três tipos de erros são somados e divididos pela quantidade de pedestres, resultando na taxa de erro por anotação.

Já o MOTP é calculado por

$$MOTP = 1 - \frac{d_{err}}{n_{matches}}, \tag{5.2}$$

onde d_{err} é a soma das diferenças métricas entre a posição rastreada e a localização real, e $n_{matches}$ é o número de casos em que ocorreu uma correspondência entre o rastreamento e a anotação verdadeira.

Além disso, incluímos as métricas comuns de Precisão - *Precision* (Prec) e Revocação - *Recall* (Rcll) (OLSON; DELEN, 2008). Prec é dada por

$$Prec = \frac{VP}{VP + FP} \tag{5.3}$$

e Rcll é obtido por

$$Rcll = \frac{VP}{VP + FN}, \tag{5.4}$$

onde Verdadeiro Positivo (VP) é o número de detecções que correspondem a objetos das anotações verdadeiras.

5.3 COMPARAÇÃO COM O ESTADO DA ARTE

Comparamos nosso método usando o WILDTRACK com outras técnicas de estado da arte na Tabela 1. Alguns métodos têm melhor desempenho, como Cheng et al. (2023) e Nguyen et al. (2022). No entanto, eles exigem treinamento. Nosso trabalho é uma abordagem direta que não requer treinamento, sendo mais fácil de aplicar em novos ambientes.

Além disso, o nosso método é *online*, enquanto Vo et al. (2021) e Nguyen et al. (2022) são *offline*. Os algoritmos *offline* têm acesso a informações de todos os quadros para associar pedestres. Portanto, sua precisão geralmente é superior a de métodos *online*. Por outro lado, eles limitam a quantidade de possíveis aplicações, pois os resultados do rastreamento não podem ser recuperados em tempo real.

Tabela 1 – Comparação com outras técnicas de última geração no conjunto de dados WILDTRACK. Avaliamos usando os últimos 10% dos quadros para comparar com métodos supervisionados. Ong et al. (2022) é uma exceção que usou todo o conjunto de dados. O nosso é o melhor método não supervisionado.

	Técnica	Online	MOTA↑	MOTP↑	FP↓	FN↓	MM↓
Supervisionado	You e Jiang (2020)	✓	74,6%	78,9%	114	107	21
	Vo et al. (2021)		75,8%	-	-	-	-
	Cheng et al. (2023)	✓	81,6%	81,8%	-	-	-
	Nguyen et al. (2022)		97,1%	-	71	7	12
Não supervisionado	Ong et al. (2022)	✓	69,7%	73,2%	424	1333	104
	Lyra. et al. (2022)	✓	77,1%	94,77%	**76**	128	**14**
	Nosso	✓	**78,05%**	**97,90%**	119	**73**	17

Fonte: Elaborada pela autora (2024)

5.4 ESTUDO DE ABLAÇÃO

Nesta seção, apresentamos os resultados ao adicionar cada módulo proposto ao rastreador base utilizando as bases de dados WILDTRACK (Subseção 5.4.1) e MultiviewX (Subseção 5.4.2). Além disso, a Subseção 5.4.1 também traz uma avaliação variando a quantidade de câmeras utilizadas.

5.4.1 Resultados no WILDTRACK

Avaliamos a adição de cada módulo ao rastreador base. Os resultados são apresentados na Tabela 2. Primeiro, tentamos replicar a técnica de Lyra. et al. (2022), obtendo resultados razoavelmente semelhantes. Na tabela, incluímos os resultados da nossa replicação e os resultados relatados no artigo original.

Além disso, é possível ver que o GSE sozinho não melhora os resultados. Isso ocorre porque, embora diminua o número de FNs, ele aumenta os FPs ao extrapolar infinitamente. No entanto, incluindo o filtro de duplicatas, o algoritmo tem um desempenho melhor do que o rastreador base.

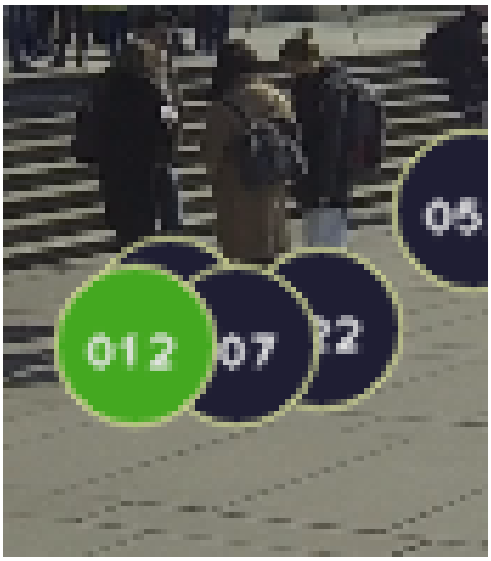
No WILDTRACK, alguns pedestres que estavam em anotações de quadros anteriores pa-

raram de ser anotados para alguns quadros. Isso faz com que uma detecção seja classificada como FP, mesmo que a pessoa ainda esteja lá. A Figura 35 mostra casos de detecções regulares (marrom) e uma detecção criada por extrapolação (verde) que contava como FP.

Figura 35 – Detecções de pedestres reais que não estão nas anotações WILDTRACK e foram classificados como FP.

(a) A detecção normal é classificada como FP (marrom).

(b) Detecção criada por extrapolação sendo classificada como FP (verde).



Fonte: Elaborada pela autora (2024)

Além disso, como o objetivo era recuperar detecções, observamos que a aplicação de GSE+Filtro reduziu FNs em 17,98%, o que confirma a hipótese H1 da Seção 1.1. A Figura 36 mostra um exemplo de recuperação bem-sucedida de FN por GSE.

Tabela 2 – Resultados de cada módulo, além do rastreador base no conjunto de dados WILDTRACK. Tentamos replicar o rastreador base e obtivemos um resultado semelhante, mas diferente. Incluímos ambos os resultados na tabela para esclarecimento. Adicionamos GSE e o filtro de duplicatas à replicação do rastreador base.

Técnica	MOTA↑	MOTP↑	Prec↑	Rcll↑	FP↓	FN↓	MM↓
Replicação de Lyra. et al. (2022)	76,89%	98,13%	88,33%	90,65%	114	89	17
Lyra. et al. (2022)	77,1%	94,77%	**91,56%**	86,55%	**76**	128	**14**
Replicação de Lyra. et al. (2022)+GSE	76,47%	97,52%	87,03%	**92,33%**	131	**73**	20
Replicação de Lyra. et al. (2022)+Filtro	76,37%	**98,15%**	88,60%	89,81%	110	97	18
Replicação de Lyra. et al. (2022)+GSE+Filtro	**78,05%**	97,90%	88,08%	**92,33%**	119	**73**	17

Fonte: Elaborada pela autora (2024)

Figura 36 – O pedestre #22 estava sendo rastreado inicialmente. No entanto, ele desapareceu no quadro seguinte. Posteriormente, o GSE conseguiu recuperar a detecção.

(a) Pedestre #22 sendo normalmente detectado no quadro anterior na câmera #3.

(b) Pedestre #22 sendo normalmente detectado no quadro anterior na câmera #1.

(c) Pedestre #22 sendo extrapolado com GSE na câmera #3.

(d) Pedestre #22 sendo extrapolado com GSE na câmera #1.

Fonte: Elaborada pela autora (2024)

Também avaliamos nosso módulo com um detector mais recente que foi proposto por Lima et al. (2023). Essa abordagem gera pseudo-rótulos usando um detector não-supervisionado e usa esses rótulos para treinar um método supervisionado. Ou seja, embora esse método exija treinamento, ele não precisa de anotações coletadas manualmente, pois elas são geradas automaticamente. Apresentamos os resultados na Tabela 3. Apenas ao substituir o método de detecção já obtivemos resultados superiores com o rastreador base. No entanto, ao adicionar GSE, o MOTA se torna ainda melhor, pois diminuímos o número de FNs e MMs. A redução de FNs reforça a hipótese H1. Contudo, a adição do filtro não influencia nos resultados.

Tabela 3 – Resultados do rastreador proposto usando as detecções mais recentes de Lima et al. (2023) na base de dados WILDTRACK.

Técnica	MOTA↑	MOTP↑	Prec↑	Rcll↑	FP↓	FN↓	MM↓
Lima et al. (2023)	78,68%	**93,77%**	**95%**	85,82%	**43**	135	25
Lima et al. (2023) + GSE	**80,25%**	91,96%	92,99%	**89,18%**	64	**103**	**21**
Lima et al. (2023) + Filter	78,68%	**93,77%**	**95%**	85,82%	**43**	135	25
Lima et al. (2023) + GSE + Filter	**80,25%**	91,96%	92,99%	**89,18%**	64	**103**	**21**

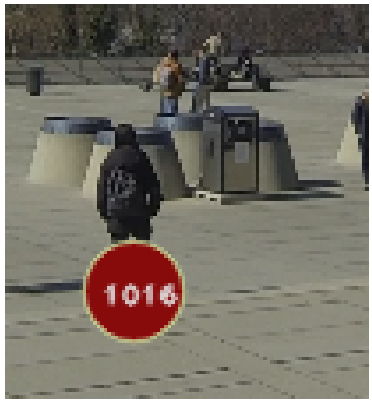
Por fim, medimos o tempo de execução da nossa técnica no WILDTRACK. GSE e filtro gastam 0,0740 segundos por quadro. Como o rastreador base leva 0,0498 segundos, o tempo total para rastrear pedestres, dadas as detecções prontas para uso, é de 0,1238 segundos por

quadro. Isso significa que o algoritmo pode rastrear pedestres a uma taxa de 8 FPS.

5.4.1.1 Variação no Número de Câmeras

A técnica usada para detecção recupera apenas os pedestres que são vistos por pelo menos duas câmeras. Portanto, há casos em que o pedestre é claramente visto de um ponto de vista, mas isso não é suficiente para recuperar sua detecção, como no caso ilustrado na Figura 37.

Figura 37 – Um pedestre que não é detectado em nosso método, apesar de estar visível, pois só aparece em uma câmera.



Fonte: Elaborada pela autora (2024)

Além disso, conduzimos experimentos variando o número de câmeras. Descobrimos que aumentar o número de câmeras é bom, mas mais câmeras não significa necessariamente melhores resultados. Além disso, usar GSE e filtros melhorou os resultados, mesmo ao usar diferentes configurações de câmera, contribuindo para a hipótese H1. Na Tabela 4, é possível ver que o MOTA usando seis câmeras é maior do que usando sete.

O motivo é que usamos várias câmeras para obter uma detecção mais precisa. Uma detecção só é recuperada se o detector a encontrou em pelo menos duas câmeras. No entanto, se ambas as câmeras tiverem pontos de vista semelhantes, um pedestre visto apenas em uma câmera é menos confiável. Portanto, só vale a pena incluir câmeras com mudanças significativas de posição. Neste caso, a câmera #7 é um pouco semelhante à câmera #1, e sua adição criou FPs que são vistos apenas nelas. Após chegar a essa conclusão, analisamos se outro par de câmeras era semelhante. Como a câmera #4 parecia a câmera #1, avaliamos usando as câmeras 1+2+3+5+6, e essa configuração obteve o melhor MOTA.

Tabela 4 – Resultados usando um número diferente de câmeras.

Câmeras	MOTA↑	MOTP↑	Prec↑	Rcll↑	FP↓	FN↓	MM↓
1+2	30,04%	93,25%	**98,76%**	33,51%	**4**	633	29
1+2 (c/ GSE+Filtro)	42,65%	90,07%	95,91%	46,85%	19	506	21
1+2+3	56,20%	91,50%	96,53%	61,34%	21	368	28
1+2+3 (c/ GSE+Filtro)	64,81%	93,85%	95,33%	70,69%	33	279	23
1+2+3+4	54,83%	90,56%	91,77%	64,39%	55	339	36
1+2+3+4 (c/ GSE+Filtro)	61,66%	90,19%	90,04%	73,11%	77	256	32
1+2+3+4+5	66,49%	90,31%	91,97%	77,00%	64	219	36
1+2+3+4+5 (c/ GSE+Filtro)	71,95%	93,94%	91,76%	81,83%	70	173	24
1+2+3+4+5+6	78,99%	97,87%	92,54%	88,55%	68	109	23
1+2+3+4+5+6 (c/ GSE+Filtro)	81,72%	97,59%	91,99%	91,70%	76	79	19
1+2+3+4+5+6+7	76,89%	**98,13%**	88,33%	90,65%	114	89	17
1+2+3+4+5+6+7 (c/ GSE+Filtro)	78,05%	97,90%	88,08%	**92,33%**	119	**73**	17
1+2+3+5+6	80,25%	97,49%	95,05%	86,66%	43	127	18
1+2+3+5+6 (c/ GSE+Filtro)	**82,77%**	97,35%	94,27%	89,92%	52	96	**16**

Fonte: Elaborada pela autora (2024)

5.4.2 Resultados no MultiviewX

Também demonstramos os resultados do nosso método no conjunto de dados MultiviewX na Tabela 5. Como o artigo original do rastreador base não mostrou resultados neste conjunto de dados, apresentamos apenas os resultados da nossa replicação.

Como esperado, detectar pedestres neste conjunto de dados é mais desafiador, pois possui mais pedestres em média do que o WILDTRACK. Há muitos FNs, mas conseguimos recuperar 40,12% deles adicionando o módulo GSE, confirmando ainda mais a hipótese H1. Mesmo que o número de FPs aumente, a recuperação de detecções é superior à geração de erros. A decisão de priorizar resultados corretos mais precisos ou coletar mais dados sobre rotas de pedestres pode ser tomada de acordo com a aplicação. Além disso, o filtro não teve impacto nos resultados.

Tabela 5 – Resultados de cada módulo, além do rastreador base no conjunto de dados MultiviewX.

Técnica	MOTA↑	MOTP↑	Prec↑	Rcll↑	FP↓	FN↓	MM↓
Replicação de Lyra. et al. (2022)	70,15%	**96,14%**	**98,54%**	76,97%	**17**	344	**85**
Replicação de Lyra. et al. (2022)+GSE	**72,49%**	90,75%	92,53%	**86,21%**	104	**206**	101
Replicação de Lyra. et al. (2022)+Filtro	70,15%	**96,14%**	**98,54%**	76,97%	**17**	344	**85**
Replicação de Lyra. et al. (2022)+GSE+Filtro	**72,49%**	90,75%	92,53%	**86,21%**	104	**206**	101

Fonte: Elaborada pela autora (2024)

6 CONSIDERAÇÕES FINAIS

Neste capítulo, apresentamos um breve resumo com as conclusões obtidas nesse trabalho na Seção 6.1. Além disso, a Seção 6.2 lista as contribuições acadêmicas. Por fim, a Seção 6.3 aborda lacunas e possíveis trabalhos futuros.

6.1 CONCLUSÕES

Neste trabalho, propusemos um módulo *online* que não requer treinamento para recuperar detecções faltantes. Usamos extrapolação linear para criar novas detecções e uma GPR para suavizar as coordenadas. Além disso, filtramos detecções próximas para remover duplicatas.

Avaliamos o desempenho do módulo proposto usando um rastreador base no conjunto de dados WILDTRACK. Isso resultou em 78,05% de MOTA, o que é uma melhoria de 0,95 p.p. e o torna o melhor método não supervisionado. Além disso, reduzimos o número de FNs em 17,98%, validando a hipótese H1.

Também avaliamos nossa solução no conjunto de dados MultiviewX, onde o MOTA foi de 72,49%. Melhoramos o rastreador base em 2,34 p.p. e reduzimos o número de FNs em 40,12%, o que também corrobora com a hipótese H1.

6.2 CONTRIBUIÇÕES

As contribuições deste trabalho foram:

- Um método online que não requer treinamento e pode ser vinculado a rastreadores multi-câmera 3D existentes para recuperar detecções ausentes;
- Um resumo extendido *RetryTRACK: Recovering Misses in Multi-camera Pedestrian Tracking* aceito no *LatinX in AI workshop* na conferência *Computer Vision and Pattern Recognition Conference* (ANDRADE; LIMA; TEICHRIEB, 2024b);
- Um artigo completo *RetryTRACK: Recovering Misses in Multi-camera 3D Pedestrian Tracking* aceito na conferência *Conference on Graphics, Patterns and Images (SIBGRAPI)* (ANDRADE; LIMA; TEICHRIEB, 2024a).

6.3 TRABALHOS FUTUROS

Embora tenhamos melhorado a precisão do rastreador, nossa técnica ainda tem a limitação de gerar alguns FPs. Além disso, em trabalhos futuros, planejamos agregar calibração automática (LEE et al., 2022; ZHANG et al., 2020) ao nosso método, pois a necessidade de câmeras calibradas também pode ser uma desvantagem para aplicações reais.

Ademais, como o número de MMs foi aumentado no MultiviewX, gostaríamos de investigar o uso da aparência de pedestres para distinguir sujeitos com IDs diferentes (HASHEMPOOR; KOIKARA; HWANG, 2024). Por fim, gostaríamos de investigar abordagens que não assumem uma distribuição gaussiana, como o filtro de partículas (AWAL et al., 2023).

REFERÊNCIAS

AHMADYAN, A.; HOU, T.; WEI, J.; ZHANG, L.; ABLAVATSKI, A.; GRUNDMANN, M. *Instant 3D Object Tracking with Applications in Augmented Reality*. 2020.

ANDRADE, I. de; LIMA, J. Multi-camera 3d pedestrian tracking using graph neural networks. In: INSTICC. *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - Volume 5: VISAPP*. [S.l.]: SciTePress, 2023. p. 974–981. ISBN 978-989-758-634-7.

ANDRADE, I. de; LIMA, J. P.; TEICHRIEB, V. Retrytrack: Recovering misses in multi-camera 3d pedestrian tracking. In: *2024 37th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2024. p. 1–6.

ANDRADE, I. de; LIMA, J. P.; TEICHRIEB, V. Retrytrack: Recovering misses in multi-camera pedestrian tracking [poster presentation]. In: *International Conference on Computer Vision: LatinX in AI (LXAI) Research Workshop 2024*. Seattle WA, USA: [s.n.], 2024.

AWAL, M. A.; REFAT, M. A. R.; NAZNIN, F.; ISLAM, M. Z. A particle filter based visual object tracking: A systematic review of current trends and research challenges. *International Journal of Advanced Computer Science & Applications*, v. 14, n. 11, 2023.

AZARCOYA-CABIEDES, W.; VERA-ALFARO, P.; TORRES-RUIZ, A.; AGUEZ, J. A.-n. S.-R. Automatic detection of bumblebees using video analysis. *DYNA*, scieloco, v. 81, p. 81 – 84, 10 2014. ISSN 0012-7353. Disponível em: <http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0012-73532014000500009&nrm=iso>.

BERGMANN, P.; MEINHARDT, T.; LEAL-TAIXE, L. Tracking without bells and whistles. In: *ICCV*. [S.l.: s.n.], 2019.

BERNARDIN, K.; STIEFELHAGEN, R. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, Springer, v. 2008, p. 1–10, 2008.

BEVILACQUA, V.; MAIO, A. D. A computer vision and control algorithm to follow a human target in a generic environment using a drone. In: HUANG, D.-S.; HAN, K.; HUSSAIN, A. (Ed.). *Intelligent Computing Methodologies*. Cham: Springer International Publishing, 2016. p. 192–202. ISBN 978-3-319-42297-8.

BEWLEY, A.; GE, Z.; OTT, L.; RAMOS, F.; UPCROFT, B. Simple online and realtime tracking. In: *ICIP*. [S.l.: s.n.], 2016. p. 3464–3468.

BREZINSKI, C.; ZAGLIA, M. R. *Extrapolation methods: theory and practice*. [S.l.]: Elsevier, 2013.

CAO, J.; PANG, J.; WENG, X.; KHIRODKAR, R.; KITANI, K. Observation-centric sort: Rethinking sort for robust multi-object tracking. In: *CVPR*. [S.l.: s.n.], 2023. p. 9686–9696.

CHAVDAROVA, T.; BAQUÉ, P.; BOUQUET, S.; MAKSAI, A.; JOSE, C.; BAGAUTDINOV, T.; LETTRY, L.; FUA, P.; GOOL, L. V.; FLEURET, F. Wildtrack: A multi-camera hd dataset for dense unscripted pedestrian detection. In: *CVPR*. [S.l.: s.n.], 2018. p. 5030–5039.

CHENG, C.-C.; QIU, M.-X.; CHIANG, C.-K.; LAI, S.-H. Rest: A reconfigurable spatial-temporal graph model for multi-camera multi-object tracking. In: *ICCV*. [S.l.: s.n.], 2023. p. 10051–10060.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: IEEE. *CVPR*. [S.l.], 2005. v. 1, p. 886–893.

DU, Y.; ZHAO, Z.; SONG, Y.; ZHAO, Y.; SU, F.; GONG, T.; MENG, H. Strongsort: Make deepsort great again. *IEEE TMM*, v. 25, p. 8725–8737, 2023.

FELZENSZWALB, P.; MCALLESTER, D.; RAMANAN, D. A discriminatively trained, multiscale, deformable part model. In: IEEE. *CVPR*. [S.l.], 2008. p. 1–8.

GALIL, Z. Efficient algorithms for finding maximum matching in graphs. Association for Computing Machinery, New York, NY, USA, v. 18, n. 1, p. 23–38, mar. 1986. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/6462.6502>.

HASHEMPOOR, H.; KOIKARA, R.; HWANG, Y. D. *FeatureSORT: Essential Features for Effective Tracking*. 2024. Disponível em: <https://arxiv.org/abs/2407.04249>.

HERNANDEZ, D. A. M.; NALBACH, O.; WERTH, D. How computer vision provides physical retail with a better view on customers. In: IEEE. *2019 IEEE 21st Conference on Business Informatics (CBI)*. [S.l.], 2019. v. 1, p. 462–471.

HOU, Y.; ZHENG, L.; GOULD, S. Multiview detection with feature perspective transformation. In: *ECCV*. [S.l.: s.n.], 2020.

KALMAN, R. E. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, v. 82, n. 1, p. 35–45, 03 1960. ISSN 0021-9223. Disponível em: <https://doi.org/10.1115/1.3662552>.

KARP, R. M. *Reducibility among Combinatorial Problems*. Boston, MA: Springer US, 1972. 85–103 p. ISBN 978-1-4684-2001-2. Disponível em: <https://doi.org/10.1007/978-1-4684-2001-2_9>.

KIM, A.; OšEP, A.; LEAL-TAIXé, L. Eagermot: 3d multi-object tracking via sensor fusion. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. [S.l.: s.n.], 2021. p. 11315–11321.

KUBALE, M. *Graph colorings*. [S.l.]: American Mathematical Soc., 2004. v. 352.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, n. 11, p. 2278–2324, 1998.

LEE, S.-E.; SHIBATA, K.; NONAKA, S.; NOBUHARA, S.; NISHINO, K. Extrinsic camera calibration from a moving person. *IEEE Robotics and Automation Letters*, v. 7, n. 4, p. 10344–10351, 2022.

LEI, M.; SONG, Y.; ZHAO, J.; WANG, X.; LYU, J.; XU, J.; YAN, W. End-to-end network for pedestrian detection, tracking and re-identification in real-time surveillance system. *Sensors*, v. 22, n. 22, 2022. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/22/22/8693>.

LI, J.; WANG, C.; ZHU, H.; MAO, Y.; FANG, H.-S.; LU, C. Crowdpose: Efficient crowded scenes pose estimation and a new benchmark. In: *CVPR*. [S.l.: s.n.], 2019. p. 10863–10872.

LIMA, J. a. P.; ROBERTO, R.; FIGUEIREDO, L.; oES, F. S.; THOMAS, D.; UCHIYAMA, H.; TEICHRIEB, V. 3d pedestrian localization using multiple cameras: a generalizable approach. *Mach. Vision Appl.*, Springer-Verlag, Berlin, Heidelberg, v. 33, n. 4, jul 2022. ISSN 0932-8092. Disponível em: <https://doi.org/10.1007/s00138-022-01323-9>.

LIMA, J. P.; ROBERTO, R.; FIGUEIREDO, L.; SIMOES, F.; TEICHRIEB, V. Generalizable multi-camera 3d pedestrian detection. In: *CVPR*. [S.l.: s.n.], 2021. p. 1232–1240.

LIMA, J. P.; THOMAS, D.; UCHIYAMA, H.; TEICHRIEB, V. Toward unlabeled multi-view 3d pedestrian detection by generalizable ai: Techniques and performance analysis. In: *SIBGRAPI*. [S.l.: s.n.], 2023. p. 1–6.

LIU, W.; ANGUELOV, D.; ERHAN, D.; SZEGEDY, C.; REED, S.; FU, C.-Y.; BERG, A. C. Ssd: Single shot multibox detector. In: SPRINGER. *ECCV*. [S.l.], 2016. p. 21–37.

LU, Y.; MA, H.; SMART, E.; YU, H. Real-time performance-focused localization techniques for autonomous vehicle: A review. *IEEE Transactions on Intelligent Transportation Systems*, v. 23, n. 7, p. 6082–6100, 2022.

LYRA., V.; de Andrade., I.; LIMA., J.; ROBERTO., R.; FIGUEIREDO., L.; TEIXEIRA., J.; THOMAS., D.; UCHIYAMA., H.; TEICHRIEB., V. Generalizable online 3d pedestrian tracking with multiple cameras. In: INSTICC. *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP,*. [S.l.]: SciTePress, 2022. p. 820–827. ISBN 978-989-758-555-5. ISSN 2184-4321.

NEGRI, P.; GARAYALDE, D. Pedestrian tracking using probability fields and a movement feature space. *Dyna*, 2006, Revista DYNA, v. 84, n. 200, p. 217–227, 2017.

NGUYEN, D. M. H.; HENSCHEL, R.; ROSENHAHN, B.; SONNTAG, D.; SWOBODA, P. Lmgp: Lifted multicut meets geometry projections for multi-camera multi-object tracking. In: *CVPR*. [S.l.: s.n.], 2022. p. 8866–8875.

OLSON, D. L.; DELEN, D. *Advanced data mining techniques*. [S.l.]: Springer Science & Business Media, 2008. 138 p.

ONG, J.; VO, B.-T.; VO, B.-N.; KIM, D. Y.; NORDHOLM, S. A bayesian filter for multi-view 3d multi-object tracking with occlusion handling. *IEEE TPAMI*, v. 44, n. 5, p. 2246–2263, 2022.

REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. You only look once: Unified, real-time object detection. In: *CVPR*. [S.l.: s.n.], 2016. p. 779–788.

REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. *arXiv*, 2018.

SHIWAKOTI, N.; SARVI, M.; BURD, M. Using non-human biological entities to understand pedestrian crowd behaviour under emergency conditions. *Safety Science*, v. 66, p. 1–8, 2014. ISSN 0925-7535. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925753514000186>.

STANOJEVIC, V. D.; TODOROVIC, B. T. Boosttrack: boosting the similarity measure and detection confidence for improved multiple object tracking. *Machine Vision and Applications*, Springer, v. 35, n. 3, p. 1–15, 2024.

SUN, Z.; CHEN, J.; CHAO, L.; RUAN, W.; MUKHERJEE, M. A survey of multiple pedestrian tracking based on tracking-by-detection framework. *IEEE TCSVT*, IEEE, v. 31, n. 5, p. 1819–1833, 2020.

VO, M.; YUMER, E.; SUNKAVALLI, K.; HADAP, S.; SHEIKH, Y.; NARASIMHAN, S. G. Self-supervised multi-view person association and its applications. *IEEE TPAMI*, v. 43, n. 8, p. 2794–2808, 2021.

WANG, J. An intuitive tutorial to gaussian process regression. *Computing in Science & Engineering*, v. 25, n. 4, p. 4–11, 2023.

WANG, Y.-H.; HSIEH, J.-W.; CHEN, P.-Y.; CHANG, M.-C.; SO, H.-H.; LI, X. Smiletrack: Similarity learning for occlusion-aware multiple object tracking. *AAAI*, v. 38, n. 6, p. 5740–5748, Mar. 2024. Disponível em: <https://ojs.aaai.org/index.php/AAAI/article/view/28386>.

WOJKE, N.; BEWLEY, A.; PAULUS, D. Simple online and realtime tracking with a deep association metric. In: *ICIP*. [S.l.: s.n.], 2017. p. 3645–3649.

YI, K.; LUO, K.; LUO, X.; HUANG, J.; WU, H.; HU, R.; HAO, W. Ucmctrack: Multi-object tracking with uniform camera motion compensation. *AAAI*, v. 38, n. 7, p. 6702–6710, Mar. 2024. Disponível em: <https://ojs.aaai.org/index.php/AAAI/article/view/28493>.

YOU, Q.; JIANG, H. Real-time 3d deep multi-camera tracking. *arXiv preprint arXiv:2003.11753*, arXiv, 2020.

ZHANG, A.; LIPTON, Z. C.; LI, M.; SMOLA, A. J. *Dive into Deep Learning*. [S.l.]: Cambridge University Press, 2023. <https://D2L.ai>.

ZHANG, C.; RAMEAU, F.; KIM, J.; ARGAW, D. M.; BAZIN, J.-C.; KWEON, I. S. Deepptz: Deep self-calibration for ptz cameras. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. [S.l.: s.n.], 2020.

ZHANG, Y.; SUN, P.; JIANG, Y.; YU, D.; WENG, F.; YUAN, Z.; LUO, P.; LIU, W.; WANG, X. Bytetrack: Multi-object tracking by associating every detection box. In: AVIDAN, S.; BROSTOW, G.; CISSÉ, M.; FARINELLA, G. M.; HASSNER, T. (Ed.). *ECCV*. Cham: Springer Nature Switzerland, 2022. p. 1–21. ISBN 978-3-031-20047-2.

ZHOU, H.; HU, H. Human motion tracking for rehabilitation—a survey. *Biomedical signal processing and control*, Elsevier, v. 3, n. 1, p. 1–18, 2008.

ZHOU, X.; KOLTUN, V.; KRÄHENBÜHL, P. Tracking objects as points. In: VEDALDI, A.; BISCHOF, H.; BROX, T.; FRAHM, J.-M. (Ed.). *ECCV*. Cham: Springer International Publishing, 2020. p. 474–490. ISBN 978-3-030-58548-8.