Generalizable Online 3D Pedestrian Tracking with Multiple Cameras

Victor Lyra[1][a], Isabella de Andrade[2], João Paulo Lima[2,1][b], Rafael Roberto[1], Lucas Figueiredo[3,1][c], João Marcelo Teixeira[4,1], Diego Thomas[5][d], Hideaki Uchiyama[6][e] and Veronica Teichrieb[1][f]

[1] *Voxar Labs, Centro de Informática, Universidade Federal de Pernambuco, Recife, Brazil*
[2] *Departamento de Computação, Universidade Federal Rural de Pernambuco, Recife, Brazil*
[3] *Unidade Acadêmica de Belo Jardim, Universidade Federal Rural de Pernambuco, Belo Jardim, Brazil*
[4] *Departamento de Eletrônica e Sistemas, Universidade Federal de Pernambuco, Recife, Brazil*
[5] *Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan*
[6] *NARA Institute of Science and Technology, Nara, Japan*

Keywords: Tracking, Detection, Multiple Cameras, Pedestrians.

Abstract: 3D pedestrian tracking using multiple cameras is still a challenging task with many applications such as surveillance, behavioral analysis, statistical analysis, and more. Many of the existing tracking solutions involve training the algorithms on the target environment, which requires extensive time and effort. We propose an online 3D pedestrian tracking method for multi-camera environments based on a generalizable detection solution that does not require training with data of the target scene. We establish temporal relationships between people detected in different frames by using a combination of graph matching algorithm and Kalman filter. Our proposed method obtained a MOTA and MOTP of 77.1% and 96.4%, respectively on the test split of the public WILDTRACK dataset. Such results correspond to an improvement of approximately 3.4% and 22.2%, respectively, compared to the best existing online technique. Our experiments also demonstrate the advantages of using appearance information to improve the tracking performance.

1 INTRODUCTION

Tracking a great number of pedestrians in a large area is a challenging task that has received a lot of attention in the reserach community because of its potential application to security applications, surveillance, and behavioral analysis. Standard RGB cameras are often used to monitor large urban areas because of their low price, easy (re-)deployment and wide range of application possibilities. As a consequence, the standard problem is how to track pedestrians from one (or multiple) 2D video(s) of a target scene.

Pedestrian tracking solutions can use single or multiple cameras. Using a single camera brings challenges such as imprecise 3D estimation, difficulty in covering a large area, crowd tracking, and occlusions, resulting in information loss and can produce ambiguity on trajectory interpretation. On the other hand, using multiple cameras with overlapping views makes 3D tracking possible even in these challenging scenarios. In addition, the available area for tracking is increased, and it is easier to deal with occlusions since different cameras can better view the occluded persons.

By its turn, tracking pedestrians with multiple cameras leads to other challenges. For instance, it is necessary to identify individuals consistently through multiple views, requiring a higher computational power given that more images are used as input. Therefore, many existing solutions that use this approach were developed to work offline (Zhu, 2019), meaning that all available data is used as input at once after all observed events occur. It opposes online solutions that process data as they are obtained, being able to work in real-time and provide helpful information for interactions with ongoing activities on the monitored area.

[a] https://orcid.org/0000-0003-3508-3486
[b] https://orcid.org/0000-0002-1834-5221
[c] https://orcid.org/0000-0001-9848-5883
[d] https://orcid.org/0000-0002-8525-7133
[e] https://orcid.org/0000-0002-6119-1184
[f] https://orcid.org/0000-0003-4685-3634

Lyra, V., de Andrade, I., Lima, J., Roberto, R., Figueiredo, L., Teixeira, J., Thomas, D., Uchiyama, H. and Teichrieb, V.
Generalizable Online 3D Pedestrian Tracking with Multiple Cameras.
DOI: 10.5220/0010842800003124
In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2022) - Volume 5: VISAPP*, pages 820-827
ISBN: 978-989-758-555-5; ISSN: 2184-4321

Furthermore, many state-of-the-art solutions that use deep learning techniques have to train on the target dataset (Vo et al., 2020). The consequence is that the solution has to be trained again in every change of scenario. In other words, it is not generalizable. Moreover, such a training procedure often demands significant efforts to annotate ground-truth data and a considerable amount of processing time.

In this context, we propose an online and generalizable 3D pedestrian tracking solution based on a generalizable multi-camera detection solution (Lima et al., 2021) and a multi-camera tracking approach inspired by the SORT single-camera tracking algorithm (Bewley et al., 2016).

Our tracking solution follows the *tracking-by-detection* paradigm (Sun et al., 2020b). In this approach, pedestrians are independently detected in each frame. These detections are connected with detections of other frames according to their proximity or similarity, tracing each person's trajectory. The tracking procedure takes place on the ground plane, where we assign identities (IDs) to each pedestrian, and these IDs are reassigned to the same pedestrians at a subsequent time. Thus, the routes that each pedestrian traveled are formed from the previous locations of the same ID.

Our contributions are:

- An approach for tracking pedestrian locations on the ground plane from a generalizable multi-camera detection solution that adapts the SORT algorithm to track the ground points while considering multi-view pedestrian appearance (Section 3);
- Comparisons with different configurations of the distance between pedestrians and the use of histograms or re-identification networks (Section 4);
- Quantitative and qualitative evaluations using an in-the-wild dataset regarding the proposed method's tracking performance concerning state-of-the-art multi-camera 3D tracking approaches (Section 4).

2 RELATED WORK

Human tracking as a multi-object tracking (MOT) problem has been intensively studied. Generally, existing tracking approaches belong in one of two categories: single-camera or multi-camera.

2.1 Single-camera Tracking

The literature has primarily discussed the problem of MOT applied over a set of images from a single camera. For instance, the MOTChallenge benchmark (Leal-Taixé et al., 2015) has provided many MOT challenges each year in different scenes and configurations, such as moving or stationary cameras, and a ranking of the best solutions for each challenge.

There are many tracking-by-detection methods for single-camera tracking. Some examples are the SORT algorithm, a solution that uses a Kalman filter to model the movement of the tracked objects (Bewley et al., 2016), and Deep SORT, which uses association metrics through a re-identification neural network to improve correspondence reliability between detections at different times (Wojke et al., 2017).

Among the state-of-the-art solutions listed in the MOTChallenge benchmark, convolution neural networks (CNNs) are used in graphs to create an association between persons in different time instants and therefore track them (Papakis et al., 2020). For example, the work of Zhou et al. (2020) used neural networks to estimate the offset of each person in an image, tracking them throughout the video (Zhou et al., 2020).

These methods aim to solve the MOT problem using only one view of the scene, whereas our approach uses multiple views from different angles.

2.2 Multi-camera Detection and Tracking

As mentioned in Section 1, using multiple cameras has advantages such as better dealing with occlusions and allowing 3D tracking. However, it can increase the required computational power since it uses more images.

You and Jiang (2020) is an example of this approach, where a neural network is used to directly estimate the location of pedestrians in a ground plane from images for later accomplishing tracking using bipartite graphs (You and Jiang, 2020). The work of Vo et al. (2021) uses almost the entire dataset to train a person descriptor with unsupervised learning, thus identifying each pedestrian separately (Vo et al., 2020). Finally, the work of Ong et al. (2020) uses a generalizable Multi-Bernoulli filter (Vo et al., 2016) integrated with a Bayesian recursion jointly with a new system for treating occlusions in multi-camera environments (Ong et al., 2020).

Some solutions do not fuse detections before tracking. For example, Chen et al. (2016) and Sun et al. (2020) propose to execute tracking for each cam-

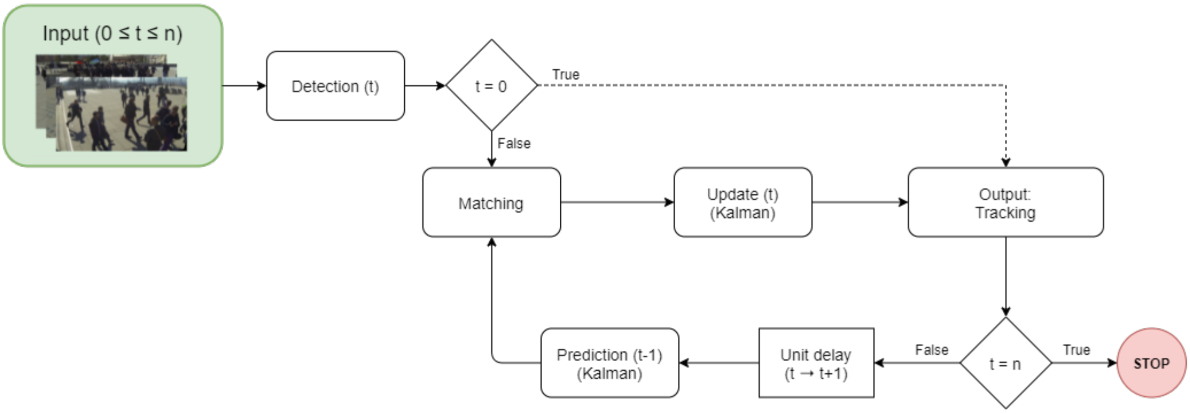


Figure 1: Diagram with the pipeline of our tracking solution, with *t* being the time and *n* the number of images in the video.

era individually and later combine the routes of each pedestrian in a 3D environment (Chen et al., 2016; Sun et al., 2020a).

However, the problem of tracking using multiple cameras is still restricted in the literature. Most solutions for this type of problem use deep learning to detect and track. So, to obtain good results, it is necessary to train in the target dataset previously. Therefore, most of these solutions do not present a good result in a change of scenario unless previous training is performed.

On the other hand, a more generalizable solution was proposed using deep learning to estimate pedestrian skeletons and subsequently using the estimated heel joints to calculate the ground points for each pedestrian in each camera (Lima et al., 2021). It is unnecessary to perform training in the target dataset for this kind of estimation, therefore being more generalizable. These points are projected in the same ground plane, and a graph is created where detections of different cameras are connected. The positions of pedestrians are calculated through a clique clover algorithm[1]. However, this solution works without calculating a relation between different times; thus, it does not perform tracking, just the detection of pedestrians.

Our solution uses this generalizable detection to track the pedestrians, making our solution generalizable compared to the other solutions. It also works in an online manner, while some others only work with offline inputs.

3 METHOD

To maximize the accuracy and consistency of our solution, we used matching algorithms in graphs, Kalman filter, histograms, and re-identification networks. Differently from the SORT algorithm, we used the Kalman filter on the ground points of the pedestrians and incorporated multi-camera pedestrian appearance into the solution. Figure 1 shows the pipeline of our method.

3.1 Detection

Before tracking pedestrians using multiple cameras, it is necessary to detect them in each camera and fuse these detections in each frame. For that, we choose a generalizable detection method that calculates pedestrians' 3D locations in the ground plane (Lima et al., 2021). Figure 2 shows two input images with their detections and the ground plane with locations of each detected pedestrian.

However, as Figure 3 shows, this method of calculating 3D pedestrian locations does not maintain consistency between detection identities of different time instants.

Besides the location, for each instant *t*, we collect additional information like the color histogram and the person identification features for each detection.

3.1.1 Re-identification Features

Person re-identification (re-ID) is the problem of recognizing individuals that appeared on different cameras or in the same camera but on different occasions. It is a challenging task due to the presence of different viewpoints, varying low-image resolutions, illumination changes, and many others (Ye et al., 2021).

[1] https://en.wikipedia.org/wiki/Clique_cover

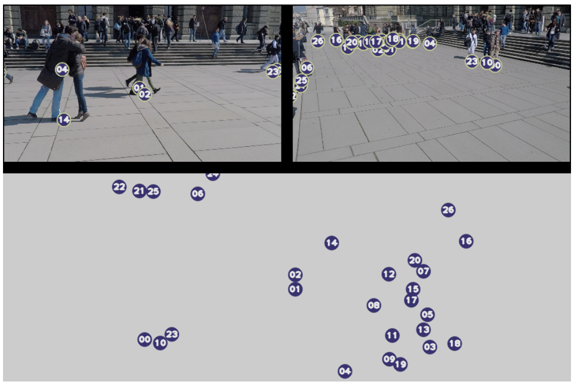
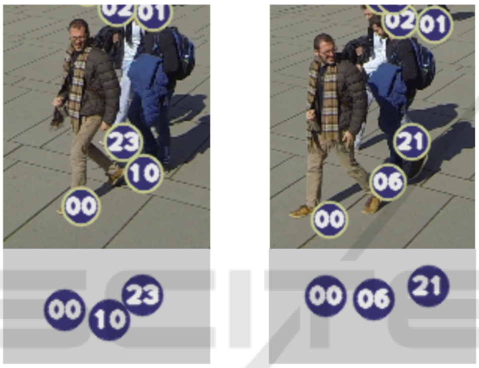


Figure 2: Images of two different cameras that were captured at the same time with its detections and, at the bottom, the ground plane with the location of each pedestrian in the scene.



Detections in t - 1

Detections in t

Figure 3: Detection of pedestrians in subsequent times.

In our problem, images with similar tones are prone to belong to the same person. We can compare color tones between different images by calculating color histograms.

With the histograms of each detection in hand, we calculate the average histogram defined by

$$h_{avg} = \frac{\sum_{i=1}^{n} h_i}{n}, \tag{1}$$

where we sum the histograms h_i of fused detections from 1st to n-th camera, then divide by n.

On the other hand, some re-ID solutions are implemented with CNNs, although they can have difficulties dealing with significant changes in the domain (Song et al., 2019). These networks receive an image as input, execute the processing and return a *feature vector* that describes the image and should be similar for inputs of the same person. We can calculate the similarity between different inputs with a distance measure such as cosine similarity.

Therefore, for the re-ID features extracted using the pre-trained model of Deep SORT (Wojke et al., 2017), we also calculate the average described as

$$f_{avg} = \frac{\sum_{i=1}^{n} f_i}{n}, \tag{2}$$

where we sum the re-ID features f_i of fused detections from 1st to n-th camera, then divide by n.

3.2 Prediction

In order to establish temporal relationships between detections, a 4 dimension state $(x, y, \dot{x}, \dot{y})$ is defined to represent a pedestrian, considering (x, y) as the pedestrian position and $(\dot{x}, \dot{y})$ as his/her speed that is initialized as zero. Thereby, the Kalman Filter (Kalman, 1960) is applied to predict the pedestrian location in the next instant of time.

The prediction step uses the current state of the system to predict its next state

$$\hat{\mathbf{x}}_{n+1,n} = \begin{bmatrix} \hat{x}_{n+1,n} \\ \hat{\dot{x}}_{n+1,n} \end{bmatrix} = \begin{bmatrix} \hat{x}_{n,n} + \Delta t\, \hat{\dot{x}}_{n,n} \\ \hat{\dot{x}}_{n,n} \end{bmatrix} \tag{3}$$

and predict the covariance of the system. For systems with constant dynamics, we have

$$p_{n+1,n} = p_{n,n}. \tag{4}$$

3.3 Matching

Given the pedestrians predicted locations, we construct a bipartite graph, illustrated in Figure 4, where vertices are detections predictions for time $t-1$ and new detections of time t. The edges are created only between detections of different instants, and if the distance between them is less them a distance threshold d_{max} defined as follows:

$$d_{max} = \frac{v_{max}}{FPS}, \tag{5}$$

where v_{max} is the maximum speed of a pedestrian in m/s and FPS is the number of images per second of the input video.

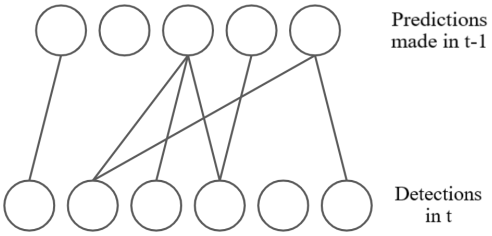


Figure 4: Example of a bipartite graph created with detections of subsequent times.

For each edge, a weight is calculated through arithmetic mean considering Euclidean distance

(d_e) between detections, Hellinger distance (d_h) (Hellinger, 1909) between histograms, and cosine similarity (d_c) between re-ID vectors, thus

$$p(i,j) = (1-\lambda_h-\lambda_c)\, d_e(i,j) + \lambda_h\, d_h(i,j) + \lambda_c\, d_c(i,j), \quad (6)$$

where i and j are the detections corresponding to graph vertices and λ_h and λ_c are the weights for histograms and re-ID vectors distance, respectively, to control the effect of each distance over the weight of graph, considering

$$0 \leq \lambda_h + \lambda_c \leq 1. \quad (7)$$

In possession of the created graph, we use an algorithm of maximum matching (Galil, 1986), which gives a subset of edges without common vertices and is maximum because it returns a subset with as many edges as possible. It also considers the weights of the edges to return a subset where the sum of matched edges weights has a minimum value.

3.4 Update

With the subset of the edges from the matching algorithm, each edge gives a pair of vertices considered as the same person. The location of the pedestrian is updated using the update step of the Kalman filter.

The update combines the prediction of the system's state with a new measurement using a weighted average. Thus, it makes values with less uncertainty have greater weight through the Kalman gain K_n:

$$K_n = \frac{P_{n,n-1}}{P_{n,n-1} + r_n}. \quad (8)$$

Finally, the estimated state and covariance are updated as follows:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + K_n(z_n - \hat{x}_{n,n-1}), \quad (9)$$

$$p_{n,n} = (1-K_n)p_{n,n-1}. \quad (10)$$

If any new detections are not matched, they are added as new pedestrians in tracking. Moreover, if any of the already tracked pedestrians is not matched, the Kalman filter is applied to predict their position, using it in the subsequent matching, overcoming temporary occlusions. Finally, if the pedestrian does not match for τ time instants, it is considered that he has left the scene, as shown in Figure 5.

Therefore, these steps are repeated for each new frame until there are no more images, for the case of a pre-recorded video.

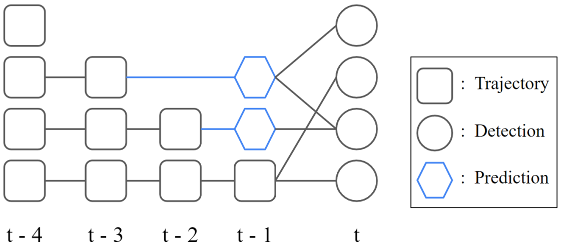


Figure 5: Representation of trajectories and the prediction of the Kalman filter being used for unmatched pedestrians for $\tau = 3$.

4 RESULTS

We performed experiments on a challenging multi-camera dataset, obtained quantitative and qualitative results, and observed the execution time. The results are presented in the following subsections.

4.1 Dataset and Metrics

The WILDTRACK dataset (Chavdarova et al., 2018), considered challenging for multiple camera tracking, was used to evaluate the proposed solution. It consists of video frames extracted from seven different cameras, recorded on a street with many people, while providing ground-truth annotations for a total of 400 frames and the camera's intrinsic and extrinsic parameters.

We used the py-motmetrics Python library to perform a quantitative evaluation (Heindl, 2017), calculating many different metrics related to the MOT problem. Among these metrics, the most important ones are Multiple Object Tracking Accuracy (MOTA) and Multiple Object Tracking Precision (MOTP) (Bernardin and Stiefelhagen, 2008).

The MOTA metric shows how consistent the trajectories are, considering mismatches, false positives, and missing trajectories. On the other hand, the MOTP metric represents the total error in the estimated position of each tracked object divided by the number of correspondences between objects across the video. Thus, it shows the ability to estimate precise positions for objects independently of keeping consistent trajectories.

4.2 Tracking Performance Evaluation

As previously mentioned in Subsection 4.1, the tests were executed using the WILDTRACK dataset. The maximum speed of the pedestrian, for equation 5, was defined as $3m/s$ since no individuals are running and the average speed of a person walking is $1.5m/s$.

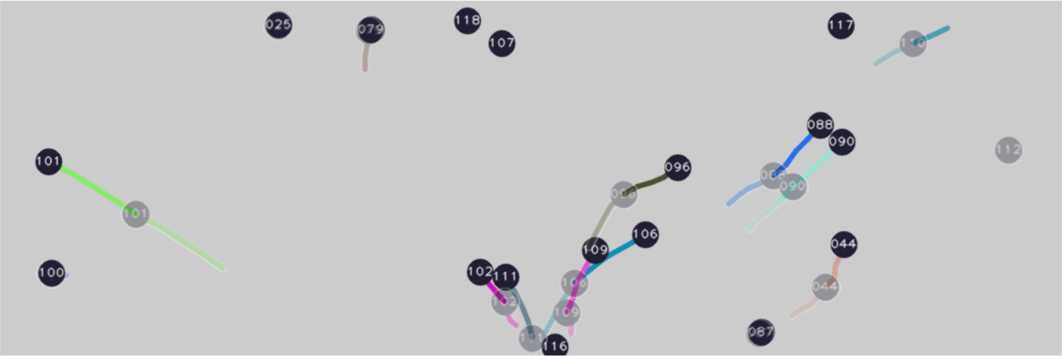


Figure 6: Pedestrian tracking results in frame 62 (with transparency) and frame 66 of the WILDTRACK dataset.

Figure 6 shows the tracking result for two frames, illustrating the location of pedestrians on a ground plane while displaying the trajectory taken by each pedestrian. In Figure 7, you can see a sequence of input frames and the respective pedestrian tracking.

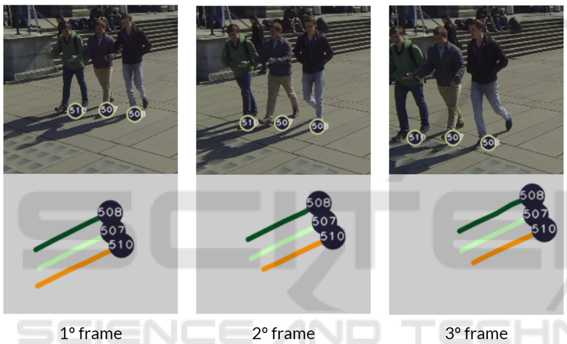


Figure 7: Accurate tracking of three pedestrians walking side by side.

A quantitative evaluation was performed considering different weight configurations (Equation 6). The results are listed in Table 1, with the best results in bold. Thereby, it is possible to understand that the greater the weight of the histogram, the worse the results become. Still, setting a small weight to the histogram or re-ID slightly improved MOTA and MOTP.

Furthermore, unlike our solution, supervised techniques have to train on the dataset. Hence the data is divided into 90% and 10% for training and testing, respectively. Therefore, the evaluation metrics were also calculated for the last 40 frames from WILDTRACK to compare it with other techniques. Table 2 exhibits our solution's MOTA and MOTP using only the distance compared to other state-of-the-art techniques for the subset from the WILDTRACK dataset. Our solution achieved the highest MOTA and MOTP being an online approach, while some competing methods are offline.

Table 1: MOTA and MOTP for different weight configurations in graph edges.

Weight Type	λ_h	λ_{id}	MOTA	MOTP
Distance	0	0	65.28%	92.21%
Distance + Histogram	0.1	0	65.26%	**93.01%**
	0.3	0	64.99%	92.77%
	0.5	0	64.90%	92.40%
	1	0	53.83%	85.41%
Distance + Re-ID	0	0.1	**65.30%**	92.92%
	0	0.3	64.74%	92.72%
	0	0.5	64.91%	92.76%
	0	1	53.75%	84.83%
Distance + Histogram + Re-ID	0.1	0.1	65.14%	92.83%
	0.2	0.2	64.88%	92.97%
	0.4	0.1	64.74%	92.38%
	0.2	0.4	64.15%	90.71%

Table 2: MOTA from the proposed solution and other techniques evaluated considering only 10% of the dataset. The solutions of (Chavdarova et al., 2018) and (Vo et al., 2020) work *offline*, while the rest work *online*.

Technique	MOTA	MOTP
(Ong et al., 2020)	69.7%	73.2%
(Chavdarova et al., 2018)	72.2%	60.3%
(You and Jiang, 2020)	74.6%	78.9%
(Vo et al., 2020)	75.8%	-
Ours	**77.1%**	**96.4%**

4.3 Evaluations with Ground-truth Data

We also evaluated our solution using the WILDTRACK ground-truth data as input to analyze the algorithm behavior when receiving perfect detections. Table 3 shows this evaluation's MOTA and MOTP using only distance and different configurations of graph weights, with the best values in bold.

Table 3: MOTA and MOTP for different weight configurations in graph edges, using ground-truth detections from WILDTRACK dataset as input for tracking.

Weight Type	λ_h	λ_{id}	MOTA	MOTP
Distance	0	0	98.39%	98.69%
Distance+ Histogram	0.2	0	98.34%	98.59%
	0.5	0	**98.87%**	**98.73%**
	1	0	97.96%	97.14%
Distance + Re-ID	0	0.2	98.41%	98.45%
	0	0.5	98.61%	98.65%
	0	1	98.28%	98.20%

With these results, it is possible to observe that, given a perfect detection, tracking has excellent accuracy and precision. Thus, the tracking algorithm is considerably dependent on the quality of detections. It is also noticeable that with the re-ID, the MOTA can increase, and when we use histograms, there is an improvement in both MOTA and MOTP metrics.

4.4 Execution Time Analysis

We analyzed the execution time of tracking without considering the time to detect the pedestrians. For each set of input detections, tracking takes approximately 49.8 milliseconds, with the code written in Python and executing in an Intel Core i7-7700HQ CPU. Thus, this type of tracking can execute in real-time, especially if rewritten in a faster programming language like C++. Table 4 discloses the execution average time of each tracking step and the total for each input in milliseconds.

Table 4: Time of total execution and each step of tracking per input in milliseconds.

	Execution Time
Matching	47.6
Prediction (Kalman)	0.8
Update (Kalman)	1.4
Total	49.8

4.5 Failure Cases

Given these results, we conducted a qualitative analysis of failure cases. For example, Figure 8 illustrates a failure case where the detection of a pedestrian that has *ID 09* in *frame #2* had a precision error, which occasioned a swap of identity between two persons in the third frame. This error was caused by the prediction step, even though both of them were walking in parallel.

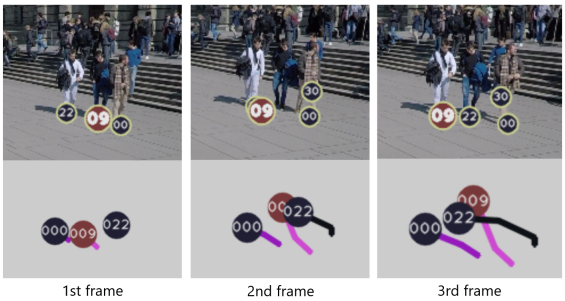


Figure 8: Failure case where a mismatch occurs.

On the other hand, Figure 9 exemplifies a failure case where a person of the group goes undetected. At the same time, a new person, without identification, appears close to this group. This error made one of the detections of the group pair with this new person during graph matching, changing the ID of a pedestrian in this group to be the ID of the new person in the scene.

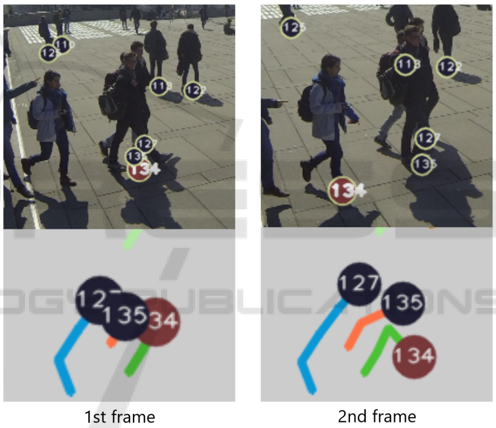


Figure 9: Failure case where the identity of a person is transferred to another.

Nevertheless, only a few trajectory errors happened when using ground-truth data of the WILDTRACK dataset as input.

5 CONCLUSIONS

This paper presented an algorithm for tracking pedestrians using multiple cameras. Our evaluation showed that it outperforms state-of-the-art methods. However, Tables 1 and 3 suggest that the accuracy of the solution for tracking proposed in this paper is mainly dependent on the quality of detection. Also, failure cases occasioned by detection errors were analyzed.

Furthermore, we also discussed the use of distances between color histograms and re-IDs to improve detection matching. We found that with the inclusion of these appearance features, both MOTA and MOTP are slightly increased.

In future work, we intend to track 2D skeletons of pedestrians in each camera (Xiu et al., 2018) instead of tracking only ground plane points to use this information to improve 3D pedestrian tracking.

ACKNOWLEDGEMENTS

The authors would like to thank Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) (process 425401/2018-9) for partially funding this research.

REFERENCES

Bernardin, K. and Stiefelhagen, R. (2008). Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10.

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE.

Chavdarova, T., Baqué, P., Bouquet, S., Maksai, A., Jose, C., Bagautdinov, T., Lettry, L., Fua, P., Van Gool, L., and Fleuret, F. (2018). Wildtrack: A multi-camera hd dataset for dense unscripted pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5030–5039.

Chen, Y., Zhao, Q., An, Z., Lv, P., and Zhao, L. (2016). Distributed multi-target tracking based on the k-mtscf algorithm in camera networks. *IEEE Sensors Journal*, 16(13):5481–5490.

Galil, Z. (1986). Efficient algorithms for finding maximum matching in graphs. *ACM Computing Surveys (CSUR)*, 18(1):23–38.

Heindl, C. (2017). py-motmetrics. https://github.com/cheind/py-motmetrics.

Hellinger, E. (1909). Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909(136):210–271.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems.

Leal-Taixé, L., Milan, A., Reid, I., Roth, S., and Schindler, K. (2015). Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*.

Lima, J. P., Roberto, R., Figueiredo, L., Simoes, F., and Teichrieb, V. (2021). Generalizable multi-camera 3d pedestrian detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1232–1240.

Ong, J., Vo, B. T., Vo, B. N., Kim, D. Y., and Nordholm, S. (2020). A bayesian filter for multi-view 3d multi-object tracking with occlusion handling. *arXiv preprint arXiv:2001.04118*.

Papakis, I., Sarkar, A., and Karpatne, A. (2020). Gcnmatch: Graph convolutional neural networks for multi-object tracking via sinkhorn normalization. *arXiv preprint arXiv:2010.00067*.

Song, J., Yang, Y., Song, Y.-Z., Xiang, T., and Hospedales, T. M. (2019). Generalizable person re-identification by domain-invariant mapping network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 719–728.

Sun, H., Chen, Y., Aved, A., and Blasch, E. (2020a). Collaborative multi-object tracking as an edge service using transfer learning. In *2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 1112–1119. IEEE.

Sun, Z., Chen, J., Chao, L., Ruan, W., and Mukherjee, M. (2020b). A survey of multiple pedestrian tracking based on tracking-by-detection framework. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(5):1819–1833.

Vo, B.-N., Vo, B.-T., and Hoang, H. G. (2016). An efficient implementation of the generalized labeled multi-bernoulli filter. *IEEE Transactions on Signal Processing*, 65(8):1975–1987.

Vo, M. P., Yumer, E., Sunkavalli, K., Hadap, S., Sheikh, Y. A., and Narasimhan, S. G. (2020). Self-supervised multi-view person association and its applications. *IEEE transactions on pattern analysis and machine intelligence*.

Wojke, N., Bewley, A., and Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE.

Xiu, Y., Li, J., Wang, H., Fang, Y., and Lu, C. (2018). Pose flow: Efficient online pose tracking. In *BMVC*.

Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., and Hoi, S. C. (2021). Deep learning for person re-identification: A survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

You, Q. and Jiang, H. (2020). Real-time 3d deep multi-camera tracking. *arXiv preprint arXiv:2003.11753*.

Zhou, X., Koltun, V., and Krähenbühl, P. (2020). Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer.

Zhu, C. (2019). Multi-camera people detection and tracking.