Rethinking Bipartite Graph Matching in Realtime Multi-object Tracking

Zhuojun Zou[1,2], Jie Hao[1,3,*] and Lin Shu[1,3]
[1]*Institute of Automation, Chinese Academy of Sciences, Beijing, China*
[2]*School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China*
[3]*Guangdong Institute of Artificial Intelligence and Advanced Computing, Guangzhou, China*
Email: {zouzhuojun2018, jie.hao, lin.shu}@ia.ac.cn

2022 Asia Conference on Algorithms, Computing and Machine Learning (CACML) | 978-1-6654-8290-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/CACML55074.2022.00124

***Abstract*—Data association is a crucial part for tracking-by-detection framework. Although many works about constructing the matching cost between trajectories and detections have been proposed in the community, few researchers pay attention to how to improve the efficiency of bipartite graph matching in realtime multi-object tracking. In this paper, we start with the optimal solution of integer linear programming, explore the best application of bipartite graph matching in tracking task and evaluate the rationality of cost matrix simultaneously. Frist, we analyze the defects of bipartite graph matching process in some multi-object tracking methods, and establish a criteria of similarity measure between trajectories and detections. Then we design two weight matrices for multi-object tracking by applying our criteria. Besides, a novel tracking process is proposed to handle visual-information-free scenario. Our method improves the accuracy of the graph-matching-based approach at very fast running speed (3000+ FPS). Comprehensive experiments performed on MOT benchmarks demonstrate that the proposed approach achieves the state-of-the-art performance in methods without visual information. Moreover, the efficient matching process can also be assembled on approaches with appearance information to replace cascade matching.**

***Keywords*-Bipartite Graph Matching, Multi-object Tracking, Tracking-by-detection**

I. Introduction

Multiple object tracking (MOT) is one of the most challenging problems in computer vision due to its wide range of applications, such as autonomous vehicle[1], [2], intelligent video surveillance[3], [4] and smart environments and ambient technologies[5], [6]. In recent years Tracking-by-Detection paradigm has become a popular tool for MOT[7], which divides the tracking process into two major parts: a **detector** generates object candidates for each frame at first, and then a **tracker** connects those detection results into multiple trajectories. As shown in Fig. 1, tracker is a crucial part of multi-object tracking approach whose main task is to (1) design matching matrix between historical trajectories and candidates, and (2) utilize matching algorithm to connect trajectories with detections according to the matrix.

Although the construction method of matching matrix changes rapidly with the improvement of computer performance and the evolution of artificial neural network structure, applying bipartite graph matching as data association method is popular and enduring[8], [9], [10]. A classic method to apply this paradigm is SORT[8], which is favored by the industry for its simple structure and extremely fast tracking speed. Following this study, DeepSORT[11] introduced visual information into matrix design and proposed a cascade process for their multi-level matching weights, thus improves the accuracy of SORT and achieves better results with the development of deep neural network[12]. We notice that both SORT and DeepSORT contain a post-process to filter matchings, and weaken the optimality of assignment problem solution. Inspired by this, we redesign the matching cost of SORT by introducing a criteria called optimal bipartite graph matching(OBGM) condition, which can be applied to any graph matching based multi-object tracker. Similarly, we also constructed a weight matrix according to the OBGM condition for DeepSORT.

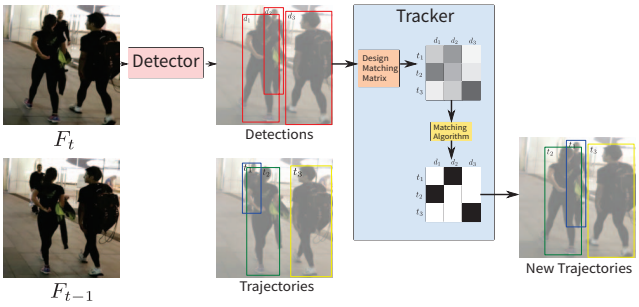

Figure 1. Illustration of the Tracking-by-Detection framework.

II. Related Work

Multi-Object Tracking There are mainly two kinds of approaches: detection-free models[13], [14], [15] and Tracking-by-Detection(TbD) models[10], [9], [11]. The formers construct an end-to-end structure by appending a task-specific branch to traditional detection network. With little help from detection datasets, the main disadvantage of this paradigm is the need for an enormous number of tracking annotation. Besides, the specific branch is limited to tracking scene, for example, the model proposed in [13] can only predict people and cars' movement, not other categories.

*Corresponding Author.
The authors declare that there is no conflict of interest regarding the publication of this paper.

978-1-6654-8290-5/22/$31.00 ©2022 IEEE
DOI 10.1109/CACML55074.2022.00124

On the contrary, TbD models enable trackers to take full use of the detection results, and the performance of TbD models are not lose to that of detection-free models on varies benchmarks[16], [17]. In TbD paradigm, the detector is trained on large-scale detection datasets which completely separate from training process of the tracker. The low coupling of the two modules makes it convenient for the detector and tracker to combine freely according to diverse application requirements.

Data association in MOT Network flow based models[18], [19] obtain the global data association solution by modeling the whole video sequence as a graph and introducing a virtual source point and sink point, but are inappropriate for real-time requirements. Multiple Hypothesis Trackers[20], [21] build a hypothesis tree to enumerate combinations of associations and maximize the a-posteriori probability by path search, and have the problem of high computational complexity. Bipartite graph matching is widely used in filter based trackers[8], [11], which constructs continuous tracking results by obtaining the optimal matchings of trajectories and detections frame by frame, and a careful design of the matching cost is necessary. Betke and Wu[22] and Luo et. al.[23] reviewed the data association methods in multi-object tracking task.

Bipartite graph matching in SORT process Simple online and realtime tracking(SORT)[8] implements probabilistic inference as a Kalman filter and data association as bipartite graph matching. The method takes Intersection-over-Union(IoU) as the similarity measure of trajectories and detections, and achieve considerable accuracy with high frame rate. Following this study, DeepSORT[11] introduced appearance features and proposed a cascade matching strategy to improve SORT by reducing the number of identity switches. Besides in a recent research, a method named SmartSORT[24] was constructed to learn the cost matrix of bipartite graph based on DeepSORT. Although many methods improved data association in multi-object tracking from the perspective of matrix construction, the matching efficiency drew limit attention. In this paper, we focus on the effectiveness of bipartite graph matching in realtime multi-object tracking, and not only successfully improves the performance of data association but also verifies the correctness of matching matrix by utilizing the properties of optimal solution.

III. Proposed Methods

In this section, we first introduce the optimal bipartite graph matching in Sec. III-A, then propose an efficient matching approach based on SORT framework in Sec. III-B including a redesigned weight matrix following the optimization criteria mentioned in Sec. III-A and a handling algorithm for real-time multi-object tracking, and finally construct a matching process in DeepSORT framework in Sec. III-C.

A. Optimal Bipartite Graph Matching

The association of trajectory set T(indices $\mathcal{T} = \{1, \ldots, N\}$) and detection set D(indices $\mathcal{D} = \{1, \ldots, M\}$) can be described as a complete bipartite graph matching problem which aims to maximize a Integer Linear Program(ILP).

Let $[w_{ij}]$ be the matching weight matrix. In SORT[8] tracking process, to ensure tracking quality, an additional gate function b_{ij} related to w_{ij} is multiplied by the optimal solution of bipartite graph matching to filter matchings. The final result is defined as

$$z^s = \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{D}} b_{ij} w_{ij} x^s_{ij} \tag{1}$$

with

$$\begin{cases} w_{ij} = iou_{ij} & (2) \\ b_{ij} = \mathbb{1}[iou_{ij} \geq t^{(1)}] & (3) \end{cases}$$

where x_s is defined as:

$$x_s = \underset{x}{\operatorname{argmax}} \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{D}} w_{ij} x_{ij}$$

$$s.t. \quad \begin{cases} \sum_{j \in \mathcal{D}} x_{ij} \leq 1 & \forall\ i \in \mathcal{T} \\ \sum_{i \in \mathcal{T}} x_{ij} \leq 1 & \forall\ j \in \mathcal{D} \\ x_{ij} \in \{0, 1\} & \forall\ i \in \mathcal{T}\ \forall\ j \in \mathcal{D} \end{cases} \tag{4}$$

where boolean variable x_{ij} denotes whether w_{ij} is selected or not. Moreover for $i \in \mathcal{T}$, $j \in \mathcal{D}$, iou_{ij} is given by

$$iou_{ij} = \frac{Intersection(T_i,\ D_j)}{Union(T_i,\ D_j)} \tag{5}$$

This metric synthetically describes the similarity of the shape and position of two bounding boxes.

The major drawback of Eq. 1 is the harmful effect of abandoned x^s_{ij} selected by b_{ij}, weakening the optimality of assignment problem solution, and z^s must be less than or equal to the result of directly solving ILP with b_{ij} constraint, which can be formulated as follow:

$$z = \max \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{D}} b_{ij} w_{ij} x_{ij}$$

$$s.t. \quad \begin{cases} \sum_{j \in \mathcal{D}} x_{ij} \leq 1 & \forall\ i \in \mathcal{T} \\ \sum_{i \in \mathcal{T}} x_{ij} \leq 1 & \forall\ j \in \mathcal{D} \\ x_{ij} \in \{0, 1\} & \forall\ i \in \mathcal{T}\ \forall\ j \in \mathcal{D} \end{cases} \tag{6}$$

Fig. 2 shows a case that $z^s < z$.

Let the solution corresponding to z in Eq. 6 be x^*, according to the above facts, we propose a optimization criteria named optimal bipartite graph matching condition that judges whether the weight matrix is set reasonably.

Theorem 1: **Optimal Bipartite Graph Matching condition:** For a measure $f(\cdot)$ **positively** correlated with tracking effect, the cost matrix w_{ij} must be inappropriately defined when $f(x^s)$ is obviously greater than $f(x^*)$.

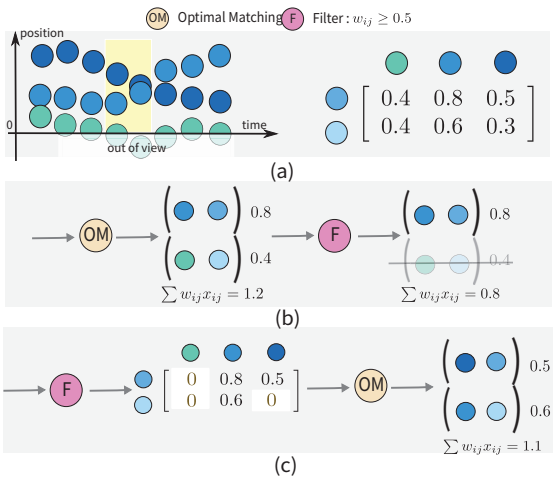

Figure 2. SORT matching (Eq. 1)and optimal matching(Eq. 6) obtain different solutions under the same weight matrix and filter setting. (a) The application of bipartite graph matching in online multi-object tracking. The weight matrix can be synthesized from multiple information sources, such as distance, visual similarity and so on, the higher is the better. (b) The process of getting z^s: filtering matching pairs to get the final result after solving the optimal matching. (c) The process of optimal matching. The method multiplies weight matrix(w_{ij}) by gate function ($b_{ij} = \mathbb{1}[w_{ij} \geq 0.5]$) to generate optimal value z. It is worth noting that $z^s \leq z$ always holds.

Replacing z^s with z will improve the performance definitely for a model with well-defined weight matrix w_{ij} and gate function b_{ij}. In order to make full use of the power of assignment problem solver such as [25], [26], w_{ij} needs to accurately reflect the relative value of matchings.

B. Efficient matching based on SORT framework

In our experiments, we observed the worse performance by replacing z_s with z when $w_{ij} = iou_{ij}$. This illustrates that iou_{ij} is a weak representation of the association between trajectories and detections according to OBGM condition. Thereby, we extend the weight matrix in SORT framework. For a tracking-by-detection process without visual information, the following weight setting is used:

$$w_{ij} = 3^{\mathbb{1}[hits_i - loss_i \geq t^{(2)}]} \times 3^{\mathbb{1}[score_j \geq t^{(3)}]} \times iou_{ij} \qquad (7)$$

The proposed w_{ij} includes three parts: the quality of tracker($3^{\mathbb{1}[hits_i - loss_i \geq t^{(2)}]}$), detection($3^{\mathbb{1}[score_j \geq t^{(3)}]}$) and matching($iou_{ij}$). Beyond that, our gating function b_{ij} is set as Eq. 3.

In Algo. 1, we developed an online multi-object tracking process with optimal bipartite graph matching solution.

To analyze the k-th frame in a video sequence, the algorithm takes historical trajectory set T_{k-1} and current detection set D_k as inputs, and outputs tracking result set R_k as well as updated trajectory set T_k. T_0 is initialized as $\emptyset$,while D_k is provided by detector.

Algorithm 1: Realtime Multi-object Tracking with Optimal Solution of Bipartite Graph Matching

Input: Detections $D = \{d_1, \ldots, d_M\}$;
Trajectories $T = \{t_1, \ldots, t_N\}$.

1 Initialize the result set $R \leftarrow \emptyset$.
2 Compute predictions $T \leftarrow \{\text{predict}(t) | t \in T\}$
3 Compute weight matrix $W = [b_{ij} w_{ij}]$
4 Compute matches $M \leftarrow$optimal_matching(W, T, D)
5 Update trajectory t with d in T, $\forall\ (t, d) \in M$
6 $T \leftarrow T \cup \{\text{new\_trajectory}(d) | d \in D \setminus \{d | (t, d) \in M\}\}$
7 $R \leftarrow \{t \mid t \in T,\ \text{max\_score}_t \geq \sigma,\ \text{hits}_t \geq L_c\}$
8 $T \leftarrow T \setminus \{t \mid\ t \in T,\ \text{loss}_t > L_{max}\ \}$
9 $T \leftarrow T \setminus$
$\{t \,| t \in T,\ \text{loss}_t > L_{min},\ \text{not } occ(t, \{d | (t, d) \in M\})\}$
10 **return** R, T

In line 2, we apply the Kalman filter proposed in [8] to predict the size and coordinate of the bounding boxes. In line 8-9, the existent time of mismatched trajectories are extended from L_{min} to L_{max} when the predicted region is occluded by other matched detections. Concretely, we formulate $occ(t, \{d_1, d_2, ...\})$ as:

$$\mathbb{1}\left[\frac{Intersection(t, Union(d_1, d_2, ...))}{Area(t)} > p\right] \qquad (8)$$

In practical applications, p is set to 80%. There is a likelihood that an object is absent intermittently for the occlusion caused by non-detected objects in realistic scenarios, which needs further analysis according to appearance features. However, the cases requiring visual aids are beyond the scope of the current discussion.

C. Efficient matching based on DeepSORT framework

DeepSORT[11] adopts a cascade matching method: It gives priority to those trajectories with younger age to match, and carries the detections without matching in this turn to the next and matches them with elder trajectories, and iterates the above process within the age range. SORT matching is applied in each iteration: filtering operation is performed after bipartite graph matching as shown in Fig. 2.

Firstly, we adjust SORT matching to our optimal matching, then we append the primary weight to the original cost matrix in the form of multiplication, and optimize the process of cascade matching to one-step matching. On this premise, we construct **minimum** weight matching matrix to retain the characteristic of DeepSORT, which is defined as follow:

$$w_{ij} = \exp(-loss_i) \times score_j \times \exp(-c_{ij}) \qquad (9)$$

where c_{ij} is a hybrid feature combined with object's appearance and position factors defined in [11]. Likewise, Eq. 9 is composed of three parts as a application of the design

Table I
E_SORT PARAMETERS IN MOT CHALLENGE.

Dataset	Detector	$t^{(1)}$	$t^{(2)}$	$t^{(3)}$	L_c	$L_{\min}$	$L_{\max}$
MOT16	FRCNN(POI)	0.2	2	0.6	1	1	3
MOT17	DPM	0.4	2	0.5	3	1	10
MOT17	SDP	0.2	1	0.6	1	1	8
MOT17	FRCNN	0.2	1	0.8	3	1	1
MOT20	FRCNN	0.4	3	0	1	10	20

Table II
COMPARISON OF EVALUATION METRICS WITH OTHER APPROACHES ON MOT16, MOT17 AND MOT20 VALIDATION SETS. * INDICATES OFFLINE TRACKERS. THE RUNNING SPEEDS OF E_SORT (·|·) REPRESENT PYTHON AND C++ APPLICATIONS RESPECTIVELY.

Method	MOTA↑	MOTP↑	MT↑	ML↓	IDS↓	FM↓	FPS↑
MOT16							
POI[27]	66.1	79.5	34.0	20.8	805	3093	10
DeepSORT[11]	61.4	79.1	32.8	18.2	781	2008	74
SmartSORT[24]	60.4	78.9	28.9	21.2	1135	2230	28
SORT[8]	59.8	79.6	25.4	22.7	1423	1835	423
E_SORT	61.1	79.2	34.7	19.6	1382	1464	363|3533
E_DeepSORT	61.4	79.1	33.6	17.8	803	2002	77
MOT17							
Tracktor++[15]	56.3	78.8	21.1	35.3	1897	3753	1.5
GMPHDOGM[31]	49.9	77.0	19.7	38.0	3125	3540	30.7
GMPHD_Rd[32]	46.8	76.4	19.7	33.3	3865	8097	30.8
IOUtracker*[33]	45.5	76.9	15.7	40.5	5988	7404	1523
SORT[8]	43.1	77.8	12.5	42.3	4852	7127	473
E_SORT	46.3	77.4	16.5	39.3	3966	4977	405|3608
MOT20							
Tracktor++[15]	52.6	79.9	29.4	26.7	1648	4374	1.2
SORT[8]	42.7	78.5	16.7	26.2	4470	17798	92
GMPHD_Rd[32]	44.7	77.5	23.6	22.1	7492	11153	25
E_SORT	45.6	77.1	32.9	19.8	4838	6071	102|530

idea mentioned in Sec III-B. Besides, b_{ij} is followed the definition in DeepSORT.

Due to the addition of visual information, the effectiveness of other auxiliary judgments unrelated to image feature is reduced. Therefore, the tracking process in DeepSORT is remain unchanged instead of using Algo. 1.

IV. EXPERIMENTAL RESULTS

A. Experiment Setting

Dataset. Our experiments are conducted on MOT16[16], MOT17 and MOT20[17] benchmarks. Those experimental datasets contain challenging videos recorded with both static and moving camera. MOT16 and MOT17 contain the same video sequences but with different detections. In the implementation on MOT16, the detections provided by POI[27] are applicated. Similarly to [11], we filter out objects with confidence score less than 0.3. The bounding boxes in MOT17 were generated by DPM[28], SDP[29], and Faster R-CNN[30], and the detector used in MOT20 is a trained Faster R-CNN.

Baseline. For a fair comparison, we rerun SORT[8] and DeepSORT[11] on the same platform as our E_SORT to test the running speed. On MOT16, our methods are compared with other approaches using the same detector, including POI, DeepSORT, SmartSORT and SORT. On MOT17 and MOT20, we provide a comparison of E_SORT with the state-of-the-art approaches without visual information[8], [33], [31], the leading neural network based method[15] and the leading GMPHD filter based method[32].

Evaluation Metrics. We assess the performance of trackers by multiple metrics including Multi-object tracking accuracy (MOTA)[34], Multi-object tracking precision (MOTP), Mostly tracked (MT>80% recovered), Mostly lost (ML<20% recovered), Identity switches (IDS), Fragmentation (FM) and speed (FPS). We regard MOTA as the main metric for its ability to measure overall tracking performance, which is expressed as:

$$MOTA = 1 - \frac{\sum_k m_k + \sum_k fp_k + \sum_k mme_k}{\sum_k g_k} \quad (10)$$

where g_k, m_k, fp_k, and mme_k are the number of present objects, misses, false positives and mismatches in the k-th frame, respectively. It combines three error sources: the ratio of missed targets($\frac{\sum_k m_k}{\sum_k g_k}$), the ratio of false positives($\frac{\sum_k fp_k}{\sum_k g_k}$) and identity switches($\frac{\sum_k mme_k}{\sum_k g_k}$). The aforementioned evaluation metric IDS proposed by Li et. al.[35] differs from this expression, which is the number of times that a trajectory changes its matched GT identity.

Implementation Details. The parameters of Kalman filter in Algo. 1 are set as SORT[8]. For the sake of simplicity, we set σ in Algo. 1 equal to $t^{(3)}$ in Eq. 7. First, we keep $L_{\min} = L_{\max} = 1$, and the remaining parameters including $t^{(1)}$, $t^{(2)}$, $t^{(3)}$, and L_c are decided by grid search. In turn, these parameters are utilized to determine $L_{\min}$ and $L_{\max}$. The specific settings are shown in Tab. I. Notice that the parameters are varied according to dataset and detection method. This is due to different trajectory number and object size in those videos, which leads to unstable performance of detectors. The annotations in MOT20 were checked manually after FRCNN detection, so threshold $t^{(3)}$ is set to 0, that is to say all detections have high reliability. Besides, L_{min} and L_{max} in Algo. 1 were much longer than videos without potential detection errors.

All experiments were run on 3.6GHz CPU. In order to give full play to the performance of E_SORT, we provides both python and C++ implementations.

B. Evaluation on MOT dataset

The comparisons with other methods are list in Tab. II. E_SORT outperforms the baseline on all three datasets, increases the MOTA of SORT by 1.3%, 3.2% and 4.7% on MOT16, MOT17 and MOT20 respectively. In particular, E_SORT substantially reduces the percentage of fragments by 20.2%, 30.1% and 65.9% in the three datasets.

On MOT16 and MOT20 datasets, E_SORT achieves the state-of-the-art performance in methods without visual information, and is particularly suitable for industrial applications for its extremely high running speed. Though the MOTA of

Table III
COMPONENT COMPARISON BETWEEN OUR METHOD AND THE ORIGINAL SORT ON MOT17 TRAIN SET. IN PARTICULAR, WE SHOW THE TREND OF PERFORMANCE CHANGE UNDER DIFFERENT MATRIX WEIGHT SETTINGS, GREEN AND RED STANDS NEGATIVE AND POSITIVE EFFECT ON TRACKING RESULTS RESPECTIVELY(BEST VIEW IN COLOR).

Tracker	Matching	w_{ij}	MOTA↑	MT↑	ML↓	IDS↓	FM↓
SORT	Eq. 1	Eq. 2	44.8	276	654	2681	3758
	Eq. 6	Eq. 2	44.7	273	655	2737	3806
			0.2%↓	**1.1%↓**	**0.2%↓**	**2.1%↓**	**1.2%↓**
	Eq. 1	Eq. 7	45.2	288	641	2311	3479
	Eq. 6	Eq. 7	45.2	289	640	2336	3496
			0.0%-	**0.3%↑**	**0.2%↑**	**1.1%↓**	**0.5%↓**
Algo.1	Eq. 6	Eq. 7	48.4	386	562	2306	2659

E_SORT is slightly lower than that of GMPHD_OGM[31] on MOT17 datasets, the proposed method is 117 times faster than GMPHD_OGM. Compared with IOUtracker[33] which has a delayed trajectory-filtering criterion, our online algorithm outforms it on MOTA, and reduce identity switches by 33.8% and fragmentation by 32.8%. Tab. II also shows the comparison between our approach and the state-of-the-art CNN-based method[15]. The proposed method has higher real-time performance (530 FPS vs. 1.2 FPS) with less performance loss (45.6 vs. 52.6 on the MOTA score).

TbD paradigm divides MOT task into two independent parts, and as demonstrated by the experimental results on MOT16 and MOT17, it inevitably leads to tracking effect relying heavily on the quality of detections. Moreover, in real-time tracking, a direct consequence is the linear superposition of running time in the two parts. Fast data association ensures practical running speed and leaves more room for detection. Take the efficient detection method FastYOLO[36] (155 FPS) as an example: E_SORT + FastYOLO is running at 149 FPS while GMPHD_OGM + FastYOLO is running at 26 FPS.

E_DeepSORT improves the matching speed while keeping the efficiency of DeepSORT[11]. The experiment proves that cascade matching can be replaced by one-step efficient matching. According to the OBGM condition, the similar performance of the two methods verifys the rationality of the weight setting in DeepSORT.

C. Ablation Study

To validate the effects of the proposed theory and method, we replace the components in the online tracking model, including the tracking progress(*Tracker*), the matching way of bipartite graph(*Matching*), and the weight matrix(w_{ij}). The experimental results are shown in Tab. III.

OBGM Condition Validation: According to the definition in Sec. III-A, we replace SORT matching(Eq. 1) with optimal matching(Eq. 6) to verify the effectiveness of the weight matrix. With Eq. 2 as the matching weight, replacing Eq. 1 results in the performance decline of all the metrics, which shows Eq. 2 satisfys the OBGM condition and has room for improvement. The same experiment conducted on Eq. 7 illustrates our matrix does not meet the condition, indicates Eq. 7 has good similarity measurement capability, although there is no visual information for guiding matching. In the other hand, the improvement of tracking results can been observed directly after replacing w_{ij}. Taking Eq. 6 as an example, the promotion on (MOTA /MT /ML /IDS /FM) is (1.1% /5.9% /2.3% /14.7% /8.1%) respectively. This proves the correctness of the OBGM condition and the rationality of our weight design method.

Notably, Using Algo. 1 significantly improved the tracking efficiency, and exceeds the SORT tracking progress by 33.6% and 23.9% on MT and FM respectively. Algo. 1 improves the overall performance by dynamically increasing the existent time of mismatch trajectories and substantially solved the frequent identity switch problem in SORT framework.

V. CONCLUSION

In this paper, we propose a general bipartite graph matching process in multi-target tracking task to make full use of the solver of Integer Linear Program. Besides, we construct two matching matrices for online multi-object tracking and improve the tracking process of SORT by introducing a novel handling algorithm. Sufficient experiments conducted on MOT challenge benchmarks demonstrate that the optimal solution of assignment problem can improve the efficiency of data association according to the OBGM condition. Moreover, our E_SORT outperforms the baseline at high speed and achieves the state-of-the-art performance in methods without visual information, and our one-step matching process E_DeepSORT can replace cascade matching with high tracking efficiency.

Acknowledgements This work was supported in part by the National Science and Technology Major Project from Minister of Science and Technology under Grant No. 2018AAA0103100, the Guangdong Provincial Key Research and Development Plan under Grant No. 2019B090917009, the Strategic Priority Research Program of Chinese Academy of Science under Grant No. XDB32070203, and the National Key Research and Development Program of China under Grant No. 2020AAA0105900.

REFERENCES

[1] W. Zong, C. Zhang, Z. Wang, J. Zhu, and Q. Chen, "Architecture design and implementation of an autonomous vehicle," *IEEE Access*, vol. 6, pp. 21 956–21 970, 2018.

[2] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 712–733, 2021.

[3] M. Zabłocki, K. Gościewska, D. Frejlichowski, and R. Hofman, "Intelligent video surveillance systems for public spaces–a survey," *Journal of Theoretical and Applied Computer Science*, vol. 8, no. 4, pp. 13–27, 2014.

[4] P. L. Venetianer and H. Deng, "Performance evaluation of an intelligent video surveillance system a case study," *Computer Vision and Image Understanding*, vol. 114, no. 11, pp. 1292–1302, 2010, special issue on Embedded Vision.

[5] N. Thakur and C. Y. Han, "An ambient intelligence-based human behavior monitoring framework for ubiquitous environments," *Information*, vol. 12, no. 2, 2021.

[6] ——, "Multimodal approaches for indoor localization for ambient assisted living in smart homes," *Information*, vol. 12, no. 3, 2021.

[7] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, "Multiple object tracking: A literature review," *Artificial Intelligence*, vol. 293, p. 103448, 2021.

[8] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *ICIP*. IEEE, 2016, pp. 3464–3468.

[9] S. Guo, J. Wang, X. Wang, and D. Tao, "Online multiple object tracking with cross-task synergy," in *CVPR*, June 2021.

[10] P. Chu, J. Wang, Q. You, H. Ling, and Z. Liu, "Transmot: Spatial-temporal graph transformer for multiple object tracking," 2021.

[11] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *ICIP*. IEEE, 2017, pp. 3645–3649.

[12] S. Karthik, A. Prabhu, and V. Gandhi, "Simple unsupervised multi-object tracking," *arXiv preprint arXiv:2006.02609*, 2020.

[13] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "Fairmot: On the fairness of detection and re-identification in multiple object tracking," *arXiv preprint arXiv:2004.01888*, 2020.

[14] Z. Lu, V. Rathod, R. Votel, and J. Huang, "Retinatrack: Online single stage joint detection and tracking," in *CVPR*, 2020, pp. 14 668–14 678.

[15] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without bells and whistles," in *ICCV*, 2019, pp. 941–951.

[16] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *arXiv preprint arXiv:1603.00831*, 2016.

[17] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé, "Mot20: A benchmark for multi object tracking in crowded scenes," *arXiv preprint arXiv:2003.09003*, 2020.

[18] G. Brasó and L. Leal-Taixé, "Learning a neural solver for multiple object tracking," in *CVPR*, 2020, pp. 6247–6257.

[19] P. Lenz, A. Geiger, and R. Urtasun, "Followme: Efficient online min-cost flow tracking with bounded memory and computation," in *ICCV*, 2015, pp. 4364–4372.

[20] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *ICCV*, 2015, pp. 4696–4704.

[21] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong, "Enhancing detection model for multiple hypothesis tracking," in *CVPRw*, 2017, pp. 18–27.

[22] M. Betke and Z. Wu, "Data association for multi-object visual tracking," *Synthesis Lectures on Computer Vision*, vol. 6, no. 2, pp. 1–120, 2016.

[23] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T. Kim, "Multiple object tracking: A literature review," *Artificial Intelligence*, p. 103448, 2020.

[24] M. Meneses, L. Matos, B. Prado, A. de Carvalho, and H. Macedo, "Learning to associate detections for real-time multiple object tracking," *arXiv preprint arXiv:2007.06041*, 2020.

[25] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[26] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, no. 4, pp. 325–340, 1987.

[27] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan, "Poi: Multiple object tracking with high performance detection and appearance feature," in *ECCV*. Springer, 2016, pp. 36–42.

[28] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *T-PAMI*, vol. 32, no. 9, pp. 1627–1645, 2009.

[29] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers," in *CVPR*, 2016, pp. 2129–2137.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *T-PAMI*, vol. 39, no. 6, pp. 1137–1149, 2016.

[31] Y. M. Song, K. Yoon, Y. C. Yoon, K. C. Yow, and M. Jeon, "Online multi-object tracking framework with the gmphd filter and occlusion group management," *arXiv preprint arXiv:1907.13347*, 2019.

[32] N. L. Baisa, "Occlusion-robust online multi-object visual tracking using a gm-phd filter with a cnn-based re-identification," *arXiv preprint arXiv:1912.05949*, 2019.

[33] E. Bochinski, V. Eiselein, and T. Sikora, "High-speed tracking-by-detection without using image information," in *AVSS*. IEEE, 2017, pp. 1–6.

[34] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, pp. 1–10, 2008.

[35] Y. Li, C. Huang, and R. Nevatia, "Learning to associate: Hybridboosted multi-target tracker for crowded scene," in *CVPR*, 2009, pp. 2953–2960.

[36] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016, pp. 779–788.