TrackFlow: Multi-Object Tracking with Normalizing Flows

Gianluca Mancusi[1] Aniello Panariello[1] Angelo Porrello [1] Matteo Fabbri[2]
Simone Calderara[1] Rita Cucchiara[1,3]

[1]University of Modena and Reggio Emilia, Italy [2]GoatAI S.r.l. [3]IIT-CNR, Italy

Abstract

The field of multi-object tracking has recently seen a renewed interest in the good old schema of tracking-by-detection, as its simplicity and strong priors spare it from the complex design and painful babysitting of tracking-by-attention approaches. In view of this, we aim at extending tracking-by-detection to multi-modal settings, where a comprehensive cost has to be computed from heterogeneous information e.g.*, 2D motion cues, visual appearance, and pose estimates. More precisely, we follow a case study where a rough estimate of 3D information is also available and must be merged with other traditional metrics (*e.g.*, the IoU). To achieve that, recent approaches resort to either simple rules or complex heuristics to balance the contribution of each cost. However,* i) *they require careful tuning of tailored hyperparameters on a hold-out set, and* ii) *they imply these costs to be independent, which does not hold in reality. We address these issues by building upon an elegant probabilistic formulation, which considers the cost of a candidate association as the* negative log-likelihood *yielded by a deep density estimator, trained to model the conditional joint probability distribution of correct associations. Our experiments, conducted on both simulated and real benchmarks, show that our approach consistently enhances the performance of several tracking-by-detection algorithms.*

1. Introduction

Real-time multi-person tracking in crowded real-world scenarios is a challenging and difficult problem with applications ranging from autonomous driving to visual surveillance. Indeed, the work done to create a reliable tracker that can function in every environment is noteworthy.

The most successful methods currently available in literature can be broadly grouped into three main categories: *tracking-by-detection* [10, 5, 39], *tracking-by-regression* [26, 36, 51], and *tracking-by-attention* [56, 85, 83]. In *tracking-by-detection*, bounding boxes are computed independently for each frame and associated with tracks in subsequent steps. *Tracking-by-regression* unifies detection and motion analysis, with a single module that simultaneously locates the bounding boxes and their displacement w.r.t. the previous frame. Finally, in *tracking-by-attention*, an end-to-end deep tracker based on self-attention [79] manages the life-cycle of a set of track predictions through the video sequence.

Although the two latter paradigms have recently sparked the research interest, *tracking-by-detection* still proves to be competitive [88, 11], under its simplicity, reliability, and the emergence of super-accurate object detectors [27]. In light of these considerations, we aim to strengthen *tracking-by-detection* algorithms by enriching the information they usually leverage – *i.e.*, the displacement between estimated and actual bounding boxes [82, 88] – with additional cues. Indeed, as shown by several works of multi-modal tracking [14, 87], the visual domain is just one of the possible sources that may contribute. The pose of the skeleton [16], the depth maps [62, 18] and even thermal measurements [45] are concepts that can gain further robustness, as they encode a deeper understanding of the scene. In particular, as humans move and interact in a three-dimensional space, one of the goals of this work is to provide the tracker with the (predicted) distance from the camera, thus resembling what is generally acknowledged as "2.5D". To achieve that, we train a per-istance distance deep regressor on MOTSynth [25], a recently released synthetic dataset displaying immense variety in scenes, lightning/weather conditions, pedestrians' appearance, and behaviors.

However, the fusion of multi-modal representations poses a big question: how to weigh the contribution of each input domain to the overall cost? It represents a crucial step, as its design directly impacts the subsequent assignment optimization problem: in this respect, existing works resort to handwritten formulas and heuristics *e.g.*, DeepSORT [82] computes two different cost matrices and combines them through a weighted sum. Notably, the authors of [62] build upon a probabilistic formulation, which recasts the cost $c_{i,j}$ as the likelihood of the event "the i-th detection belongs to the j-th tracklet". Afterward, it is about estimating a den-

sity function on top of correct associations, termed *inliers*. Although these fusing approaches may appear reasonable, they hide several practical and conceptual pitfalls:

- They introduce additional hyperparameters, which require careful tuning on a separate validation set and hence additional labeled data.
- A single choice of these hyperparameters cannot fit different scenes perfectly, as these typically display different dynamics in terms of pedestrians' motion and spatial density, the camera's position/motion, and lighting/weather conditions. Therefore, the right trade-off is likely to be scenario-dependent;
- Common approaches (*e.g*., a simple weighted summation) assume the input modalities to be independent, thus overlooking their interactions.

We propose to take into account the weaknesses mentioned above through a dedicated parametric density estimator – termed TrackFlow – tasked to summarize several input costs/displacements in a single output metric, *e.g*., the probability that a specific detection D belongs to a particular track T. As we strive to approximate the underlying conditional probability distribution $\mathcal{P}(D \in T|\ T)$ over the input costs, we borrow the estimator from the world of deep generative models, in particular from the literature of Normalizing Flow models [21, 22, 41]. In fact, these models represent a flexible and effective tool to perform density estimation. Moreover, we would like to emphasize the reliance of such a module on an additional context-level representation, which we provide in order to inform the model about scene-level peculiarities. This way, the computation of the likelihood is also conditioned on visual cues of the scene, which we assume may be unobserved during evaluation.

Extensive experiments on MOTSynth [25], MOT17 [57], and MOT20 [17] show that the naive cost metric – *i.e*., the 2D intersection between predicted and candidate bounding boxes – can be replaced by the score provided by our approach, with a remarkable performance gain in exchange.

2. Related Works

2.1. Multiple object tracking (MOT)

Since the advent of deep learning, advances in object detection [65, 66, 27, 90] drove the community towards *tracking-by-detection* [6, 82, 9, 88, 89, 53], where bounding boxes are associated with tracks in subsequent steps. Among the most successful works, Tracktor [6] pushes *tracking-by-detection* to the edge by relying solely on an object detector to perform tracking. CenterTrack [89] provides a point-based framework for joint detection and tracking based on CenterNet [24]. Similarly, RetinaTrack [53] extends RetinaNet [50] to offer a conceptually simple and efficient joint model for detection and tracking, leveraging instance-level embeddings. More recently, ByteTrack [88] further establishes this paradigm, unleashing the full potential of YOLOX [27]: notably, it uses almost every predicted detection, and not only the most confident ones.

As Transformers [79] gained popularity [23, 34, 52, 12], various attempts have been carried out to apply them to MOT. TransTrack [75] leverages the attention-based query-key mechanism to decouple MOT as two sub-tasks *i.e*., detection and association. Similarly, TrackFormer [56] jointly performs tracking and detection, with a single decoder network. Furthermore, MOTR [86] builds upon DETR [12] and introduce "track queries" to model the tracked instances in the entire video, in an end-to-end fashion.

Recently, a few attempts have been made to leverage 3D information for MOT. Quo Vadis [18] shows that forecast analysis performed in a bird's-eye view can improve long-term tracking robustness. To do so, it relies on a data-driven heuristics for the homography estimation: hence, it may suffer in presence of non-static cameras or target objects moving on multiple planes. Differently, PHALP [62] computes a three-attribute representation for each bounding box *i.e*., appearance, pose, and location. Similarly to our approach, they adopt a probabilistic formulation to compute the posterior probabilities of every detection belonging to each one of the tracklets.

MOT and trajectory forecasting As our approach computes an approximation of the true $\mathcal{P}(D \in T|\ T)$, it can be thought as a one-step-ahead trajectory predictor, thus resembling a non-linear and stochastic Kalman filter learned directly from data. In this respect, our work actually fits the very recent strand of literature [69, 18, 38, 58] that takes into consideration the possible applications of trajectory forecasting in MOT. Similarly to our approach, the authors of [69] perform density estimation over the location of the bounding box in the next time-step; however, while they rely on PixelRNN [77] and model $\mathcal{P}(D \in T|\ T)$ as a multinomial category distribution, we instead exploit a more flexible and powerful family of generative approaches, such as normalizing flow models [21, 68, 22]. Differently, Kesa *et al*. [38] propose a deterministic approach that simply regresses the next position of the bounding box, which, arguably, does not consider the stochastic and multi-modal nature of human motion. Finally, the authors of [58] show that a teacher-student training strategy helps when only a few past observations are available to the model, as hold in practice for newborn tracklets.

2.2. Distance estimation

The estimation of distances from monocular images is a crucial task for many computer vision applications and has been studied for many years [1, 29, 63, 30, 64, 47]. Classical approaches address the problem by regressing the

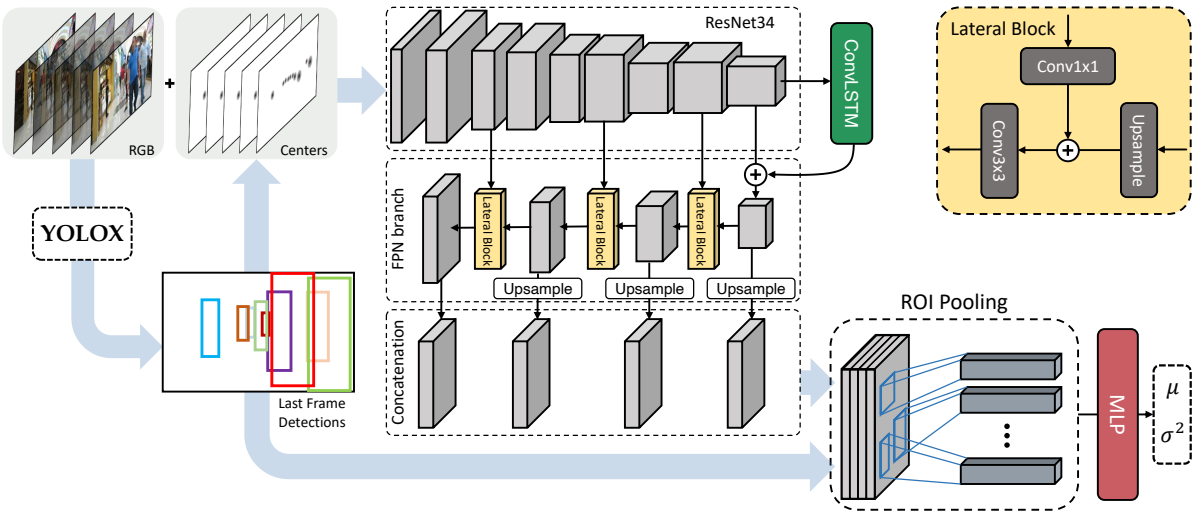


Figure 1. Overview of the camera distance estimator. DistSynth predicts per-objects distances from a short video clip. We further provide the centers of each bounding box as an additional input channel. After several convolutional blocks processing each frame independently: *i)* a temporal module is devised to extract temporal patterns; *ii)* the activation maps undergo the FPN branch, in order to preserve local details. Finally, feature maps from distinct layers are stacked and passed to the RoI pooling layer. The latter produces per-pedestrian vector representations, which we finally use to predict the pedestrians' expected distance μ and uncertainty σ^2.

relation between the object's geometry and the distance. Among these, the most popular is the inverse perspective mapping (IPM) algorithm [55, 67], which converts image points into bird's-eye view coordinates in an iterative manner. Unfortunately, due to the presence of notable distortion, IPM may yield unreliable results for objects located far from the camera. Following methods [33, 31] exploit the size of the bounding box to infer the geometry of the object; despite their simplicity and relative effectiveness, these approaches are unreliable if target instances are from different classes (*e.g.*, vehicles, people, *etc.*), or in the case of high intra-class variance (as hold for the class person).

More recently, Zhu *et al.* [91] exploit Deep Convolutional Neural Networks (CNNs) to enrich the representation of each bounding box with visual cues. In details, they firstly extract the detections through Faster R-CNN [66]; afterward, they feed a tailored CNN with the whole frame and finally apply Region of Interest (RoI) pooling, which spatially aggregates activation maps and outputs one representation for each retrieved detection. As discussed in the following section, overlapping objects may lead to erroneous predictions, as the features of foreground objects may contaminate the activations of the occluded ones.

3. Method

Our architecture comprises of two main building blocks:

- A deep neural regressor that, given a monocular image, estimates the distance of each pedestrian from the camera (see Sec. 3.1). We called it **DistSynth**, as we train only on synthetic images from MOTSynth [25].
- A deep density estimator, termed **TrackFlow** (Sec. 3.2), which has to merge 2D cues (*e.g.*, the spatial displacement between bounding boxes) with the 3D localization information obtained through DistSynth.

3.1. DistSynth: estimating per-instance distance from a monocular image

As the output of the distance estimator is meant to further refine the association cost between detections and tracks, it is crucial to handle possible temporary occlusions and noisy motion patterns. Therefore, as deeply discussed in the following, our model integrates time information (*e.g.*, a short collection of past frames) with visual cues to achieve a smoother and more reliable distance prediction.

In details, the network is fed with a short video clip $\mathbb{R}^{T\times C\times W\times H}$, where T is the length of the video clip, C is the number of channels, W and H are the width and height of the frames. As we are not interested in a dense prediction for the entire scene but in pedestrian-level predictions, we ask the network to focus its attention on a restricted set of locations: namely, in proximity of the bounding boxes provided by an off-the-shelf detector, *e.g.*, YOLOX [27]. To do so, we concatenate an additional channel to the RGB frames, representing the center of each bounding box.

The architecture mainly follows the design of residual networks [35] (in our experiments, we used ResNet-34 pretrained on ImageNet [19]). Importantly, we apply two modifications to the feature extractor to enhance its capabilities, discussed in the following two sub-paragraphs.

Exploiting temporal information While related

works [91, 33] focus solely on the last frame of interest, we propose to condition the predictions of camera distances on a small window of previous frames, thus encompassing temporal dynamics. The main goal is to provide a much more robust prediction when the target pedestrian is partially or temporarily occluded in the current frame but visible in the previous ones. In that case, his/her history would compensate and smooth the prediction. Therefore, we equip the backbone with a layer capable of processing the sequence of past feature maps: precisely, a Convolutional Recurrent Neural Network, *i.e.*, a ConvLSTM [73], whose output is a single-frame feature map encoding all the history of past frames. We insert such a module in the deeper layer of the network *i.e.*, to the exit of the last residual block of our backbone.

Improving spatial representations Standard CNNs usually exploit pooling layers to progressively down-size the activation maps. For classification, there are few doubts that such an operation provides advantageous properties (*e.g.*, translation invariance, high-level reasoning, *etc.*). Considered the task of per-object distance estimation, instead, we argue that the reliance on pooling layers could be detrimental. In fact, considered people located far away from the camera (*i.e.*, those surrounded by tiny bounding boxes), pooling layers could over-subsample the corresponding spatial regions, with a significant loss in terms of visual cues.

To avoid such a detrimental issue, we equip the feature extractor with an additional branch based on Feature Pyramid Network (FPN) [49]. In practice, it begins with the encoding produced by the temporal module, then it proceeds in the reverse direction (*i.e.*, from deepers layer to the one closer to the input), and restores the original resolution through up-sampling layers and residual paths from the forward flow. To gain a deeper understanding, Fig. 1 proposes a comprehensive visual of the architecture.

Output and loss function Once the feature maps have been processed through the temporal module and the pyramid, we again exploit the bounding boxes and perform RoI pooling [28] to obtain a feature vector for each pedestrian. The result is a $\mathbb{R}^{N\times H\times K\times K}$ feature map, where N indicates the number of detected pedestrians, H the number of hidden channels, and $K = 4$ the dimension of the RoI pooling window. We process these feature maps through a multi-layer perceptron (MLP), which outputs the predicted distances. Finally, we do not make a punctual estimate but ask the network to place a Gaussian distribution over the expected distance, thus obtaining the model's aleatoric uncertainty [8, 20]. In practice, it translates into yielding two values, $d \equiv d_\mu$ and d_{σ^2}, and optimizing the Gaussian Negative Log Likelihood (GNLL) [59], as follows:

$$\mathrm{GNLL}(d_{true}|d, d_{\sigma^2}) = \frac{1}{2}\left(\log(d_{\sigma^2}) + \frac{(d - d_{true})^2}{d_{\sigma^2}}\right).$$

3.2. TrackFlow: modeling the density of correct associations through Normalizing Flows

3.2.1 Problem statement

In a nutshell, the *tracking-by-detection* paradigm usually relies on the Kalman filter [74, 9] to estimate the next 2D spatial position $\mathbf{p}_j^{t+1} = [x_j^{t+1}, y_j^{t+1}]$ of a certain pedestrian j in the next $t+1$-th frame. The prediction $\hat{\mathbf{p}}_j^{t+1}$ depends upon the set of previous observations, contained in a short track $T_j = [\mathbf{p}_j^t, \mathbf{p}_j^{t-1}, \dots, \mathbf{p}_j^{t-|T|+1}]$ recording the past matched locations of the pedestrian j. Afterward, given a new set of detections $D_i = [\mathbf{p}_i, \mathbf{w}_i, \mathbf{h}_i]\ \ i = 1, 2, \dots, |D|$ (with $\mathbf{w}_i$ and $\mathbf{h}_i$ being the width and the height of the bounding box respectively), the *cost* of a candidate association between D_i and the track T_j can be computed as the displacement $\Delta_p \equiv \Delta_p(T_j, D_i)$ between the predicted location and the candidate one, *i.e.*, $\Delta_p = \mathrm{d}(\hat{\mathbf{p}}_j^{t+1}, \mathbf{p}_i)$. In such a notation, $\mathrm{d}(\cdot,\cdot)$ stands for any function penalizing such a displacement, as the Euclidean distance $||\hat{\mathbf{p}}_j^{t+1} - \mathbf{p}_i||_2^2$. Similarly, the variation of the sizes of the bounding box, *i.e.*, $\Delta_{w,h} \equiv \Delta_{w,h}(T_j, D_i)$ could be taken into account.

Furthermore, by virtue of the regressor introduced in Sec. 3.1, we could additionally exploit the displacement beliefs/reality relating to camera distances $\Delta_d = \mathrm{d}(\hat{d}_j^{\,t+1}, d_i)$, given the estimated one-step-ahead distance $\hat{d}_j^{\,t+1}$ for the track T_j and the distance d_i of the candidate detection D_i, inferred through DistSynth. To ease the notation, from now on we will denote T_j / D_i as T / D.

Once we have computed these costs (but other could be profitably envisioned), we shall define an aggregated cost function $\Phi(T, D) = f(\Delta_p, \Delta_{w,h}, \Delta_d)$ that jointly computes the cost of the candidate association $D \in T$. There are several approaches to achieve that; among those, we build upon the probabilistic formulation proposed in [62] and define the cost Φ as negative log-(conditional) likelihood:

$$\begin{aligned}\Phi(T, D) &= -\log \mathcal{P}_\theta(D \in T \mid T) \\ &= -\log f([\Delta_p, \Delta_{w,h}, \Delta_d] \mid T, \theta).\end{aligned}$$

In that formulation, the target conditional probability distribution $\mathcal{P}_\theta(\cdot)$ parametrizes as a learnable function $f(\cdot \mid \theta)$, where its parameters θ have to be sought by maximizing the likelihood of correct associations (often referred to as *inliers*). To ease the optimization, the authors of [62] factorized the above-mentioned density, assuming that each of the marginal distributions is independent, such that $\mathcal{P}_\theta(D \in T \mid T) \propto \mathcal{P}_p\ \mathcal{P}_{w,h}\ \mathcal{P}_d$. Therefore:

$$\Phi(T, D) = -\log \mathcal{P}_{\theta_1}(\Delta_p) - \log \mathcal{P}_{\theta_2}(\Delta_{w,h}) - \log \mathcal{P}_{\theta_3}(\Delta_d).$$

As discussed in the next subsection, we do not impose such an assumption but approximate, via Maximum Likelihood Estimation (MLE), the joint conditional distribution with a deep generative model $f(\cdot \mid T, \theta)$.

3.2.2 Overview of the architecture

Among many possible choices (*e.g.*, variational autoencoders [42], generative adversarial networks [32] or the most recent diffusion models [37]), we borrow the design of $f(\cdot|T,\theta)$ from the family of normalizing flow models [21, 68, 22]. Notably, they provide an exact estimate of the likelihood of a sample, in contrast with other approaches that yield a lower bound (as the variational autoencoder and its variants [78, 76]). Moreover, normalizing flow models grant high flexibility, as they do not rely on a specific approximating family for the posterior distribution. The latter is instead a peculiar trait of the variational methodology, which may suffer if the approximating family does not contain the true posterior distribution.

Briefly, a normalizing flow model creates an invertible mapping between a simple factorized base distribution with known density (*e.g.*, a standard Gaussian in our experiments) and an arbitrary, complex and multi-modal distribution, which in our formulation is the conditional distribution $\mathcal{P}(D \in T \mid T)$ underlying correct associations. The mapping between the two distributions is carried out through a sequence of L invertible and differentiable transformations $g_l(\cdot \mid T)$ (with parameters θ_l, omitted in the following), which progressively refines the initial density through the rule for change of variables. In formal terms, our proposal named **TrackFlow** takes the following abstract form:

$$f([\Delta_p, \Delta_{w,h}, \Delta_d] \mid T, \theta) = g_L^{-1} \circ \cdots \circ g_2^{-1} \circ g_1^{-1}, \quad (1)$$

where

$$\begin{aligned} \text{forward pass}: &\quad \mathbf{z}_l = g_l(\mathbf{z}_{l-1} \mid T);\ \mathbf{z}_L \sim \mathcal{P}_\theta(D \in T \mid T) \\ \text{inverse pass}: &\quad \mathbf{z}_{l-1} = g_l^{-1}(\mathbf{z}_l \mid T);\ \mathbf{z}_0 \sim \mathcal{N}(0,\,1) \end{aligned}$$

are the forward pass (*i.e.*, used when sampling) and the inverse pass (*i.e.*, used to evaluate densities) of TrackFlow. The model can be learned via Stochastic Gradient Descent (SGD), by minimizing the negative log-likelihood on a batch of associations sampled from the true $\mathcal{P}(D \in T \mid T)$ (*i.e.*, corresponding to valid tracks). The loss function exploits the inverse pass and takes into account the likelihood under the base distribution [44] plus an additive term for each change of variable occurred through the flow.

Base architecture Regarding the design of each layer $g_l(\cdot \mid T)$, we make use of several well-established building blocks, such as normalization layers, masked autoregressive layers [60], and invertible residual blocks [13]. In particular, our model features a cascade of residual flows [4], which we preferred to other valuable alternatives (*e.g.*, RealNVP [22]) in light of their expressiveness and proven numerical stability. For the sake of conciseness we are omitting the inverse functions, but the overall representation of the forward pass of the l-th block proceeds as follows:

$$\begin{aligned} \text{residual block}: &\quad z = \mathrm{MLP}_l(\mathbf{z}_{l-1}) + \mathbf{z}_{l-1}, \\ \text{act. norm}: &\quad z = s_l \odot z + b_l, \\ \text{masked auto. flow}: &\quad \mathbf{z}_l = \mathrm{MAF}_l(\mathrm{concat}[z \,||\, \mathrm{e}_l]), \end{aligned}$$

where e_l refers to an auxiliary learnable representation discussed below, by which we take into account the dependence on the external context (*e.g.*, the track T).

3.2.3 Context encoder

Dependence on temporal cues As stated by Eq. 1, the inverse pass (but also the forward one) of TrackFlow depends also on the observed track T. By introducing such a conditioning information, the model could learn to assign higher likelihood to the candidate associations that exhibit motion patterns coherent with those observed in the recent past. To introduce such an information, we take inspiration from [81, 71] and condition each invertible layer on an additional latent representation e_l. The latter is given by a tailored temporal encoder network e_{θ_l} s.t. $e_l = \mathrm{e}_{\theta_l}(T)$ fed with the observed track T.

Importantly: *i)* as advocated in several recent works [70, 3], we provide the encoder network with relative displacements between subsequent observations, and not with the absolute coordinates of previous positions; *ii)* regarding the design of the encoder network, it could be any module that extracts temporal features (*e.g.*, Gated Recurrent Units (GRU) [15, 71] or Transformers [79, 58]). In this work, the layout of the context encoder is a subset of the Temporal Fusion Transformer (TFT) [48], a well-established and flexible backbone for time-series analysis/forecasting. In particular, we started from the original architecture and discarded the decoding modules, employing only the layers needed to encode the previous time-steps (referred as "*past inputs*" in the original paper) of the track T.

Dependence on scene-related visual information Importantly, one of the main issues we aim to address is the (lack of) adaptation to the scene under consideration. In this respect, existing approaches devise the same aggregated cost function for all conditions, which we argue may clash with the conditions we expect in real-world settings. Indeed, since different scenarios may display substantial differences (*e.g.*, night/day, camera orientation, moving/stationary camera, *etc.*), some costs should be accordingly weighted more.

To provide such a feature, we propose to further condition the estimated density $f(\cdot|\ T, \theta)$ on a visual representation of the whole current frame. In particular, we exploit the variety of MOTSynth [25] (comprising more than five hundred scenarios) and encode each frame x^t through the CLIP's [61] visual encoder, thus profiting from its widely-known zero-shot capabilities. On top of the extracted representations, we run the k-means algorithm and split them

into $|C|= 16$ clusters, each of which represents an abstract hyper-scenario. We then introduce the index $\hat{\mathbf{c}}$ of the cluster as a further conditioning variable:

$$f \equiv f([\Delta_p, \Delta_{w,h}, \Delta_d] \mid T,\ \hat{\mathbf{c}}\ , \theta) \quad (2)$$

$$\text{where } \hat{\mathbf{c}} = \text{argmin}_{i=1,\dots,|C|}\ ||\ \text{CLIP}_v(x^t) - c_i||_2^2, \quad (3)$$

and c_i are the $|C|$ centroids retrieved through the k-means pass. Such a formulation also allows inference on novel scenarios, unseen during the training stage. To practically condition the model, we simply extend the context encoder $\text{e}_{\theta_l}(\cdot)$ to take an additional learnable embedding $\text{emb}_l[\hat{c}]$ as input, s.t. $\text{e}_l \equiv \text{e}_{\theta_l}(T, \text{emb}_l[\hat{c}])$. In practice, in light of the TFT [48] layout employed by our context encoder, it becomes natural to include scene embeddings $\text{emb}_l[\hat{c}]$ as static covariates [48, 80] – *i.e.*, something holding time-independent information about the time-series. We kindly refer the reader to the original paper [48] for all the important details regarding how the TFT uses covariates to influence the forward pass of each layer.

3.2.4 Normalization of the cost matrix

Once the density estimator $f(\cdot \mid T,\ \hat{\mathbf{c}}\ , \theta)$ has been trained, we exploit its output to fill the cost matrix $\Phi(D_j, T_i)$. Following Bastani *et al*. [2], we apply a further normalization step, defined as follows:

$$\Phi^{\text{row}} = \frac{e^{\Phi(D_j,T_i)/\sigma}}{\sum_k e^{\Phi(D_j,T_k)/\sigma}}, \qquad \Phi^{\text{col}} = \frac{e^{\Phi(D_j,T_i)/\sigma}}{\sum_k e^{\Phi(D_k,T_i)/\sigma}}, \quad (4)$$

$$\hat{\Phi}(D_j, T_i) = \min(\Phi^{\text{row}}(D_j, T_i), \Phi^{\text{col}}(D_j, T_i)). \quad (5)$$

In practice, we compute softmax (smoothed through a temperature hyperparameter σ) along rows and columns of Φ; afterward, we take the cell-wise minimum between the two cost matrices. We finally pass the normalized cost matrix $\hat{\Phi}$ to the Hungarian algorithm for solving the associations.

4. Experiments

4.1. Datasets

We evaluate the performance of our models on the MOTSynth [25], MOT17 [57], and MOT20 [17] benchmarks.

MOTSynth We train both our main models on MOTSynth [25], a large synthetic dataset designed for pedestrian detection, tracking, and segmentation in urban environments. The dataset was generated using the photorealistic video game Grand Theft Auto V and comprises 764 full-HD videos 1800 frames long recorded at 20 frames per second. The dataset includes a range of challenging scenarios, displaying various weather conditions, lighting, viewpoints, and pedestrian densities. The authors split the dataset into 190 test sequences and 574 train sequences; to select the hyperparameters of our models, we extract a holdout set of 32 sequences from the training set. To speed up the evaluation phase, we assess the tracking performance only on the first 600 frames of each testing video.

After a careful analysis of the dataset, we found a previous data-cleaning stage to be crucial. Indeed, as the dataset was collected automatically, there are several unrealistic dynamics, such as ground-truth tracks related to hidden people (*e.g.*, behind walls) for many seconds. To address it, during the tracking performance evaluation, we disable annotations for targets not visible for more than 60 frames. Eventually, we re-activate the pedestrians in case they come back into the scene. In addition, we disable the annotations for pedestrians whose distance from the camera exceeds a certain threshold (fixed at 70 meters), as we observe that they would slow down the training of the distance estimator with no evident benefits.

To better investigate the comparison of different trackers, we split the sequences of test sets based on their difficulty: namely easy, moderate, and hard. We characterize the tracking complexity through the HOTA, IDF1, and ID switches (IDs) metrics of three trackers (*i.e.*, SORT, Bytetrack, and OC-SORT) and applied *k*-means clustering to create three clusters of sequences. By doing so, the easy subset contains 26 sequences, the moderate subset contains 67 sequences, and the hard subset contains 42 sequences.

MOT17 and MOT20 We use the standard benchmarks MOT17 [57] and MOT20 [17] to evaluate multiple object tracking algorithms in crowded real-world scenarios. More specifically, MOT17 comprises seven training and seven test sequences of real-world scenarios. Similarly, the MOT20 benchmark, which is the latest addition to the MOTChallenge [17], also features challenging scenes with high pedestrian density, different indoor and outdoor locations, and different times of the day. Following the evaluation protocol introduced in [88], we define the MOT17 validation set by retaining half of its training set.

4.2. Evaluation metrics

Tracking We adhere to the CLEAR metrics [7] and provide the IDF1 and the higher-order tracking accuracy (HOTA) [54]. The former, the IDF1, evaluates the ability of the tracker to preserve the identities of objects through time and computes as the harmonic mean of the precision and recall of identity assignments between consecutive frames. The HOTA is a more comprehensive metric that simultaneously considers detection and association accuracy.

Distance estimation Besides considering the common metrics based on the squared error, we propose a novel measure, called Average Localization of Occluded objects Error (**ALOE**), tailored to measure the precision of the distance estimator for objects with a varying occlusion rate. In the following, we provide a summary of these metrics:

	Easy		Moderate		Hard		*All*	
Metrics	HOTA ↑	IDF1 ↑	HOTA ↑	IDF1 ↑	HOTA ↑	IDF1 ↑	HOTA ↑	IDF1 ↑
SORT [9]	63.48	79.40	50.31	62.11	37.48	45.13	48.42	59.05
+ TrackFlow GT	**+4.37**	**+7.41**	**+5.33**	**+9.09**	**+6.54**	**+10.88**	**+5.49**	**+9.62**
+ TrackFlow	**+0.31**	**+0.97**	**+0.81**	**+1.63**	**+0.74**	**+1.56**	**+0.54**	**+1.22**
ByteTrack [88]	63.22	80.84	49.91	62.46	37.61	46.15	48.21	59.79
+ TrackFlow GT	**+3.76**	**+2.82**	**+5.47**	**+5.51**	**+5.08**	**+4.60**	**+4.75**	**+4.54**
+ TrackFlow	**+0.13**	**+1.80**	**+0.47**	**+1.21**	**+0.88**	**+1.81**	**+0.49**	**+1.41**
OC-SORT [11]	65.56	81.61	52.42	63.50	38.10	45.48	49.96	60.16
+ TrackFlow GT	**+2.41**	**+3.76**	**+4.88**	**+7.70**	**+6.18**	**+9.55**	**+4.67**	**+7.67**
+ TrackFlow	**+0.44**	**+0.84**	**+0.60**	**+1.09**	**+1.17**	**+1.96**	**+0.31**	**+0.70**
Tracktor [6]	46.59	49.40	29.15	28.81	21.58	22.68	30.82	30.91
CenterTrack [89]	43.75	43.01	28.13	24.05	20.03	18.79	29.32	26.20

Table 1. Tracking results on MOTSynth. For each tracker, we report its extended version using either our distance estimator (*i.e.*, TrackFlow) and ground-truth distances, *i.e.*, TrackFlow (GT). For a wider comparison, we also report two *tracking-by-regression* approaches.

- **τ-Accuracy** [46] ($\delta_{<\tau}$): % of d_i *s.t.* $max(\frac{d_i}{d_i^*}, \frac{d_i*}{d_i}) = \delta < \tau$ (*e.g.*, $\tau = 1.25$), represents the maximum allowed relative error;
- **Average Localization Precision** [84, 8] ($ALP_{@\tau}$): % of d_i *s.t.* $|d_i - d_i^*| = \delta < \tau$ (*e.g.*, $\tau \in \{0.5\text{m}, 1.\text{m}, 2.\text{m}\}$) is the mean average error in true distance range;
- **Error distances** [91]: absolute relative difference (Abs. Rel.), squared relative difference (Squa. Abs.), root mean squared error (RMSE), root mean squared error in the log-space ($RMSE_{log}$);
- **$ALOE_{[\tau_1:\tau_2]}$** – Average Localization of Occluded objects Error (*ours*): avg. absolute error (meters) for objects with an occlusion level between τ_1 and τ_2, with $\tau \in [0, 1]$.

4.3. Implementations details

We feed the distance estimator with video clips of 6 frames, sampled with a uniform stride of length 8: this way, each clip lasts approximately 2 seconds. We adopt 1280×720 as input resolution, thus further preserving the visual cues. We set the batch size equal to 4 and use Adam [40] as optimizer, with a learning rate of 5×10^{-5}. On the other hand, the density estimator is trained with a batch size of 512, Adam [40] as optimizer with learning rate 1×10^{-3}. The normalizing flow consists of 16 flow blocks, each comprising 64 hidden neurons; regarding context conditioning, we fix the number of observed past observations $|T| = 8$ and the number of visual clusters $C = 16$. Unless otherwise specified, both the networks are trained only on synthetic data (*i.e.*, the training set of MOTSynth); we leave the worth-noting investigation of possible transfer learning strategies for future works.

	MOT17		MOT20	
Metrics	HOTA ↑	IDF1 ↑	HOTA ↑	IDF1 ↑
SORT [9]	64.17	72.98	60.56	74.30
+ TrackFlow	**+1.78**	**+1.41**	**+0.15**	**+0.22**
ByteTrack [88]	67.73	79.81	58.94	74.89
+ TrackFlow	**+0.40**	**+0.23**	**+0.54**	**+0.06**
OC-SORT [11]	66.22	77.74	55.18	71.22
+ TrackFlow	**+0.35**	**+1.12**	**+0.53**	**+0.76**
Tracktor++ [6]	44.66	55.00	30.36	40.63
CenterTrack [89]	48.59	58.44	31.69	41.43

Table 2. Tracking results on the validation set of MOT17 and the train set of MOT20 [17].

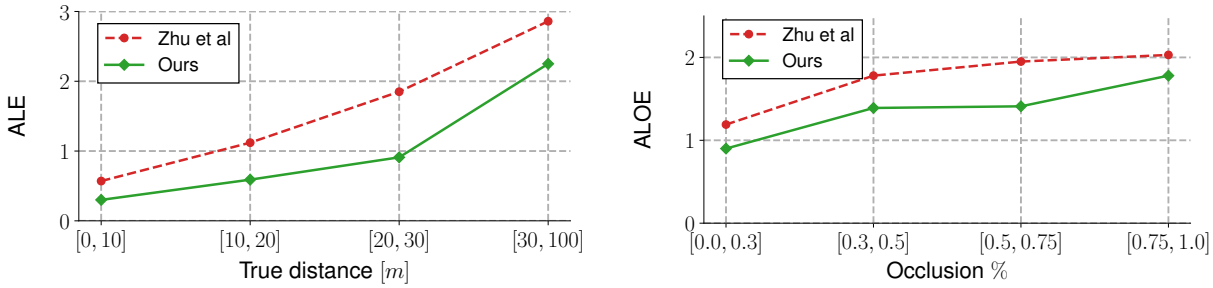
4.4. Impact on tracking-by-detection

In this section, we empirically show that our proposed method, applied to popular state-of-the-art *tracking-by-detection* techniques, improves upon the MOTSynth and the MOTChallenge benchmarks (see Tabs. 1 and 2).

On MOTSynth, we focus on three trackers (*i.e.*, SORT [9], Bytetrack [88], and OC-SORT [11]) and adhere to the following common evaluation pipeline: *i)* we compute predicted bounding boxes through YOLOX [27]; *ii)* as our approach requires an estimate of per-pedestrian camera distances, we exploit YOLOX bounding boxes by providing them to the distance estimator DistSynth (Sec. 3.1); *iii)* we finally integrate our density estimator TrackFlow into the pipeline of each tracker, applying the normalization described in Sec. 3.2.4 before the Hungarian algorithm.

			ALP ↑			ALOE ↓		
Metrics	$\delta_{<1.25}$ ↑	RMSE ↓	@0.5m	@1m	@2m	[0.3:0.5]	[0.5:0.75]	[0.75:1.]
SVR	26.7%	12.5	3.4%	6.8%	13.8%	-	-	-
DisNet [33]	27.5%	12.1	3.8%	7.5%	14.6%	-	-	-
Zhu et al. [91]	94.7%	2.15	34.5%	56.2%	78.5%	1.78	1.95	2.03
DistSynth	**99.1%**	**1.91**	**48.0%**	**68.9%**	**86.1%**	**1.39**	**1.41**	**1.78**

Table 3. Comparison of various distance estimators on MOTSynth [25]. Our DistSynth exhibits superior performance across all the metrics reported. We highlight the enhancements observed in terms of ALOE, confirming an improved ability to withstand occlusions.



(a) (b)

Figure 2. The ALE and ALOE metrics are evaluated on MOTSynth, and our method demonstrates significant improvements. Specifically, (a) our approach reduces ALE within the reported distance range shown in the plot, and (b) our method displays increased stability during occlusion events, resulting in superior performance, which can be attributed to our temporal approach.

Additionally, we provide a further comparison – termed **TrackFlow GT** (*i.e.*, *ground-truth*) – that yields an upper bound for our approach. In practice, as standard TrackFlow, it relies on YOLOX detections to compute 2D displacements, but, differently, it leverages ground-truth distances (made available in MOTSynth) in place of the DistSynth predictions. By doing so, it is possible to assess the potential of TrackFlow with near-perfect estimates.

We provide the results of such a comparison in Tab. 1. Our results indicate that TrackFlow enhances the performance of the considered trackers on all the MOTSynth splits, *i.e.*, easy, moderate, hard, highlighting the benefits of our method across three levels of complexities. In particular, the improvements are of course huge when ground-truth distances are employed; nevertheless, a consistent gain can also be appreciated when leveraging estimated distances, leading to an improved identity accuracy reflected by a steady enhancement of the IDF1 metric.

As reported in Tab. 2, the evaluation on the MOT17 and MOT20 benchmarks further shows that TrackFlow consistently improves the considered trackers in even more realistic scenarios (notably, SORT benefited the most from our approach). While evaluating on the MOT20 benchmark, we rely on the same YOLOX [27] model employed for MOT17. This particular YOLOX model was trained on two distinct datasets, namely CrowdHuman [72] and the initial half of MOT17, which aligns with the training methodology adopted in ByteTrack [88].

As mentioned before, both TrackFlow and DistSynth have been trained solely on synthetic data without additional fine-tuning, achieving still satisfying results on real data. Such a result opens the door to future research on how different components, such as the distance from the camera, can be used to advance multi-object tracking.

4.5. Distance estimation: comparison with the State-of-the-art

To assess the merits of the proposed distance estimator, we compare it with baselines and valid competitors from the current literature. We report the results of such a comparison in Tab. 3 and refer the reader to the following paragraphs for a comprehensive analysis.

Comparison with Support Vector Regressor (SVR) It consists of a simple shallow baseline based on a support vector regressor, which exploits the dimensions of the bounding boxes (*i.e.*, height and width). Through the comparison with such a naive approach, we would like to emphasize the gap w.r.t. the bias present in the task at hand *i.e.*, the smaller the bounding box, the farther the pedestrian from the camera. As expected, the SVR approach yields low performance w.r.t. our method, due to its inability to generalize to objects with different aspect ratios.

Comparison with DisNet DisNet [33] consists of an MLP of 3 hidden layers, each of 100 hidden units with SeLU [43]

		MOTSynth	MOT17		
	cond.	NLL$_\downarrow$	NLL$_\downarrow$	HOTA$_\uparrow$	IDF1$_\uparrow$
SORT [9]	-	-	-	64.17	72.98
TrackFlow	✗	-1.48	-5.66	65.34	74.77
TrackFlow	✓	$\mathbf{-1.80}$	$\mathbf{-5.81}$	**65.95**	**75.71**
TrackFlow$_{\mathrm{FT}}$	✗	-0.10	-7.29	**65.94**	75.97
TrackFlow$_{\mathrm{FT}}$	✓	$\mathbf{-0.12}$	$\mathbf{-7.50}$	65.70	**76.22**

Table 4. For MOT17, ablative study W/o scenario-level conditioning (*i.e.*, cond.) and W/o fine-tuning (*i.e.*, TrackFlow$_{\mathrm{FT}}$). Performance reported in terms of negative log-likelihood (NLL) and HOTA/IDF1 for the evaluation of the resulting tracker.

activations. The network is fed with the relative width, height, and diagonal of bounding boxes, computed w.r.t. the image dimension; these features are then concatenated with three corresponding reference values (set to 175 cm, 55 cm, and 30 cm). As can be seen, the improvements of DisNet are marginal w.r.t. SVR, but its results are substantially lower than those obtained by both Zhu *et al*. and our approach.

Comparison with Zhu *et al*. The model proposed by Zhu *et al*. [91] shares some similarities with our approach, as it relies on ResNet as feature extractor and RoI pooling to build pedestrian-level representations. However, thanks to the additional modules our model reckons on (*i.e.*, the temporal module and the FPN branch), it is outperformed by our approach under all the considered metrics. Our advancements concerning ALE and ALOE, compared to Zhu *et al*., are illustrated in Fig. 2.

4.6. Analysis of TrackFlow

We herein question the advantages of conditioning our density estimator on the scene under consideration. To do so, we focus on a single tracker (*i.e.*, SORT) and compare how its tracking performance changes if the context encoder of TrackFlow (see Sec. 3.2.3) considers only time-dependent information about the tracks and, hence, discards the scene-related visual information provided through cluster centroids c_i. From the results reported in Tab. 4 (second and third rows) it can be observed that visual conditioning (*i.e.*, the row marked with ✓) favorably leads to a lower negative log-likelihood on both the validation sets of MOTSynth and MOT17, as well as better HOTA and IDF1 results on MOT17. We interpret these findings as a confirmation of our conjectures about the advantages of designing a cost function that is aware of the scene.

Finally, we remind that only synthetic data have been used to train our models. However, it could be argued whether additional fine-tuning on real-world data could help. To shed light on this matter, we pick the best performing model attained on MOTSynth and carry out a final fine-tuning stage on the training set of MOT17, by training for further 20 epochs with lowered learning rate. We report the performance of the resulting model (*i.e.*, TrackFlow$_{\mathrm{FT}}$) W/o visual conditioning. Two major findings emerge from an analysis of the last two rows of Tab. 4: *i)* as hold for frozen models, the introduction of visual cues leads to better results (with the only exception for the HOTA on MOT17); *ii)* in general, additional training steps can profitably adapt TrackFlow to real-world scenarios, as confirmed by both the lower attained negative log-likelihood (equal to -7.50 after fine-tuning, in light of the value -5.81 prior fine-tuning) and higher tracking results.

5. Conclusion

This work presents a general approach for *tracking-by-detection* algorithms, aimed to combine multi-modal costs into a single metric. To do so, it relies on a deep generative network, trained to approximate the conditional probability distribution of *inlier costs* of correct associations. We prove the effectiveness of our approach by integrating 2D displacement and pedestrians' distances from the camera, delivered by a proposed spatio-temporal distance estimator, DistSynth, designed for crowded in-the-wild scenarios. Remarkably, our method achieves competitive results on MOTSynth, MOT17, and MOT20 datasets. Notably, we show that training solely on synthetic data yields remarkable results, indicating the importance of simulated environments for future tracking applications, especially with non-collectible real-world annotations as 3D cues. We believe our work will drive further advancements toward the exploitation of 3D clues to enhance tracking approaches in crowded scenarios.

6. Acknowledgement

The research was financially supported by the Italian Ministry for University and Research – through the PNRR project ECOSISTER ECS 00000033 CUP E93C22001100001 – and the European Commission under the Next Generation EU programme PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - "FAIR - Future Artificial Intelligence Research" - Spoke 8 "Pervasive AI". Additionally, the research activities of Angelo Porrello have been partially supported by the Department of Engineering "Enzo Ferrari" through the program FAR_2023_DIP – CUP E93C23000280005. Finally, the PhD position of Gianluca Mancusi is partly financed by Tetra Pak Packaging Solutions S.P.A., which also greatly supported the present research.

References

[1] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv preprint arXiv:1812.11941*, 2018. 2

[2] Favyen Bastani, Songtao He, and Samuel Madden. Self-supervised multi-object tracking with cross-input consistency. *Advances in Neural Information Processing Systems*, 34, 2021. 6

[3] Stefan Becker, Ronny Hug, Wolfgang Hubner, and Michael Arens. Red: A simple but effective baseline predictor for the trajnet benchmark. In *European Conference on Computer Vision Workshops*, 2018. 5

[4] Jens Behrmann, Paul Vicol, Kuan-Chieh Wang, Roger Grosse, and Jörn-Henrik Jacobsen. Understanding and mitigating exploding inverses in invertible neural networks. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021. 5

[5] Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819, 2011. 1

[6] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2019. 2, 7

[7] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 6

[8] Lorenzo Bertoni, Sven Kreiss, and Alexandre Alahi. Monoloco: Monocular 3d pedestrian localization and uncertainty estimation. In *IEEE International Conference on Computer Vision*, 2019. 4, 7

[9] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *IEEE International Conference on Image Processing*. IEEE, 2016. 2, 4, 7, 9

[10] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *14th IEEE international conference on advanced video and signal based surveillance (AVSS)*. IEEE, 2017. 1

[11] Jinkun Cao, Xinshuo Weng, Rawal Khirodkar, Jiangmiao Pang, and Kris Kitani. Observation-centric sort: Rethinking sort for robust multi-object tracking. *arXiv preprint arXiv:2203.14360*, 2022. 1, 7

[12] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision*. Springer, 2020. 2

[13] Ricky TQ Chen, Jens Behrmann, David K Duvenaud, and Jörn-Henrik Jacobsen. Residual flows for invertible generative modeling. *Advances in Neural Information Processing Systems*, 32, 2019. 5

[14] Hsu-kuang Chiu, Antonio Prioletti, Jie Li, and Jeannette Bohg. Probabilistic 3d multi-object tracking for autonomous driving. *arXiv preprint arXiv:2001.05673*, 2020. 1

[15] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Oct. 2014. 5

[16] Ross A Clark, Benjamin F Mentiplay, Emma Hough, and Yong Hao Pua. Three-dimensional cameras and skeleton pose tracking for physical function assessment: A review of uses, validity, current developments and kinect alternatives. *Gait & posture*, 68:193–200, 2019. 1

[17] Patrick Dendorfer, Aljosa Osep, Anton Milan, Konrad Schindler, Daniel Cremers, Ian Reid, Stefan Roth, and Laura Leal-Taixé. Motchallenge: A benchmark for single-camera multiple target tracking. *International Journal of Computer Vision*, 129(4):845–881, 2021. 2, 6, 7

[18] Patrick Dendorfer, Vladimir Yugay, Aljosa Osep, and Laura Leal-Taixé. Quo vadis: Is trajectory forecasting the key towards long-term multi-object tracking? *Advances in Neural Information Processing Systems*, 35, 2022. 1, 2

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. Ieee, 2009. 3

[20] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112, 2009. 4

[21] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 2, 5

[22] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 2, 5

[23] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[24] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *IEEE International Conference on Computer Vision*, 2019. 2

[25] Matteo Fabbri, Guillem Brasó, Gianluca Maugeri, Orcun Cetintas, Riccardo Gasparini, Aljoša Ošep, Simone Calderara, Laura Leal-Taixé, and Rita Cucchiara. Motsynth: How can synthetic data help pedestrian detection and tracking? In *IEEE International Conference on Computer Vision*, 2021. 1, 2, 3, 5, 6, 8

[26] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *IEEE International Conference on Computer Vision*, 2017. 1

[27] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. 1, 2, 3, 7, 8

[28] Ross Girshick. Fast r-cnn. In *IEEE International Conference on Computer Vision*, 2015. 4

[29] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017. 2

[30] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *IEEE International Conference on Computer Vision*, 2019. 2

[31] Fatih Gökçe, Göktürk Üçoluk, Erol Şahin, and Sinan Kalkan. Vision-based detection and distance estimation of micro unmanned aerial vehicles. *Sensors*, 15(9):23805–23846, 2015. 3

[32] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 5

[33] Muhammad Abdul Haseeb, Jianyu Guan, Danijela Ristic-Durrant, and Axel Gräser. Disnet: a novel method for distance estimation from monocular camera. *10th Planning, Perception and Navigation for Intelligent Vehicles (PPNIV18), IROS*, 2018. 3, 4, 8

[34] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022. 2

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016. 3

[36] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *Proceedings of the European Conference on Computer Vision*. Springer, 2016. 1

[37] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33, 2020. 5

[38] Oluwafunmilola Kesa, Olly Styles, and Victor Sanchez. Multiple object tracking and forecasting: Jointly predicting current and future object locations. In *IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, 2022. 2

[39] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *IEEE International Conference on Computer Vision*, 2015. 1

[40] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[41] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018. 2

[42] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 5

[43] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in Neural Information Processing Systems*, 30, 2017. 8

[44] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, 2020. 5

[45] Suren Kumar, Tim K Marks, and Michael Jones. Improving person tracking using an inexpensive thermal infrared sensor. In *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2014. 1

[46] Lubor Ladicky, Jianbo Shi, and Marc Pollefeys. Pulling things out of perspective. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014. 7

[47] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*, 2019. 2

[48] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021. 5, 6

[49] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017. 4

[50] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision*, 2017. 2

[51] Qiankun Liu, Qi Chu, Bin Liu, and Nenghai Yu. Gsm: Graph similarity model for multi-object tracking. In *International Joint Conference on Artificial Intelligence*, 2020. 1

[52] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *IEEE International Conference on Computer Vision*, 2021. 2

[53] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2020. 2

[54] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, 129:548–578, 2021. 6

[55] Hanspeter A Mallot, Heinrich H Bülthoff, JJ Little, and Stefan Bohrer. Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biological cybernetics*, 64(3):177–185, 1991. 3

[56] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022. 1, 2

[57] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 2, 6

[58] Alessio Monti, Angelo Porrello, Simone Calderara, Pasquale Coscia, Lamberto Ballan, and Rita Cucchiara. How many

observations are enough? knowledge distillation for trajectory forecasting. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022. 2, 5

[59] David A Nix and Andreas S Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, volume 1, pages 55–60. IEEE, 1994. 4

[60] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in Neural Information Processing Systems*, 30, 2017. 5

[61] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 2021. 5

[62] Jathushan Rajasegaran, Georgios Pavlakos, Angjoo Kanazawa, and Jitendra Malik. Tracking people by predicting 3d appearance, location and pose. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 4

[63] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *IEEE International Conference on Computer Vision*, 2021. 2

[64] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1623–1637, 2020. 2

[65] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016. 2

[66] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28, 2015. 2, 3

[67] Mahdi Rezaei, Mutsuhiro Terauchi, and Reinhard Klette. Robust vehicle detection and distance estimation under challenging lighting conditions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):2723–2743, 2015. 3

[68] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*. PMLR, 2015. 2, 5

[69] Fatemeh Saleh, Sadegh Aliakbarian, Hamid Rezatofighi, Mathieu Salzmann, and Stephen Gould. Probabilistic tracklet scoring and inpainting for multiple object tracking. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2021. 2

[70] Christoph Schöller, Vincent Aravantinos, Florian Lay, and Alois Knoll. What the constant velocity model can teach us about pedestrian motion prediction. *IEEE Robotics and Automation Letters*, 5(2):1696–1703, 2020. 5

[71] Christoph Schöller and Alois Knoll. Flomo: Tractable motion prediction with normalizing flows. In *International Conference on Intelligent Robots and Systems*. IEEE, 2021. 5

[72] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018. 8

[73] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in Neural Information Processing Systems*, 28, 2015. 4

[74] Arnold WM Smeulders, Dung M Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2013. 4

[75] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020. 2

[76] Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018. 5

[77] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International Conference on Machine Learning*. PMLR, 2016. 2

[78] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in Neural Information Processing Systems*, 30, 2017. 5

[79] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017. 1, 2, 5

[80] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*, 2017. 6

[81] Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *arXiv preprint arXiv:1912.00042*, 2019. 5

[82] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *IEEE International Conference on Image Processing*. IEEE, 2017. 1, 2

[83] Jialian Wu, Jiale Cao, Liangchen Song, Yu Wang, Ming Yang, and Junsong Yuan. Track to detect and segment: An online multi-object tracker. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2021. 1

[84] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Data-driven 3d voxel patterns for object category recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2015. 7

[85] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *IEEE International Conference on Computer Vision*, 2021. 1

[86] Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-

object tracking with transformer. In *Proceedings of the European Conference on Computer Vision*. Springer, 2022. 2

[87] Wenwei Zhang, Hui Zhou, Shuyang Sun, Zhe Wang, Jianping Shi, and Chen Change Loy. Robust multi-modality multi-object tracking. In *IEEE International Conference on Computer Vision*, 2019. 1

[88] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. In *Proceedings of the European Conference on Computer Vision*, 2022. 1, 2, 6, 7, 8

[89] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *Proceedings of the European Conference on Computer Vision*. Springer, 2020. 2, 7

[90] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2

[91] Jing Zhu and Yi Fang. Learning object-specific distance from a monocular image. In *IEEE International Conference on Computer Vision*, 2019. 3, 4, 7, 8, 9