Comparative Evaluations of Selected Tracking-by-Detection Approaches

A. A. Mekonnen and F. Lerasle

***Abstract*—In this work, we present a comparative evaluation of various *multi-person tracking-by-detection* approaches on public datasets. The work investigates popular trackers coupled with relevant visual people detectors with emphasis on exhibited performance variation depending on tracker-detector choices. Our experimental results show that tracking is sensitive to the detector choice and should be done after careful evaluation. Even a 1% difference in detector recall could lead to a 10% drop in tracking accuracy.**

***Index Terms*—Tracking-by-Detection, Multi-person Tracking, People Detection**

I. Introduction

People detection and tracking is an important research area with prominent applications in video surveillance, pedestrian protection systems, human-computer interaction, robotics, and the like. As a result, it has amassed huge interest from the scientific community as attested by recent papers [1], [2], [3]. Tracking of people in a scene, multi-person tracking, falls under multi-object tracking (MOT). MOT deals with the process of accurately estimating the state of objects – primarily, position, identity, and configuration – over time from a set of observations. Due to incurred challenges, e.g., scene clutter, target dynamics, intra/inter-class variation, measurement noise, sensor motion, and frame rate, it has long been established that coupling trackers with detectors, in a paradigm called *tracking-by-detection*, helps tackle these challenges better [4], [1], [5]. In our context, tracking-by-detection approaches rely on a people detector to start, update, re-initialize, guide (avoid drift), or terminate a tracker. In the literature, it is common to find plethora of tracking-by-detection approaches applied to people tracking. However, the usual trend is to select a single detector and directly couple it with the tracker, e.g, [1], without any comparative evaluation. With the advent of several people detection techniques that exhibit significant variations in detection performance/speed (due to remarkable advances in learning and data mining techniques)[3], [2] and the asymmetric nature of progress in detector and tracker research, this trend must change. The first step should be to evaluate the effect a detector choice has on tracking performance and relevant associations with filtering strategies therein. To the best of our knowledge no such work exists to date. There are indeed very good experimental comparative works in detection, e.g., [3], as well as in tracking, e.g., [6], but none that shows the interrelated effects of detector and tracker choices and corresponding implications in different application contexts. This paper, which is an updated and significantly extended version of our preliminary work in [7], tries to bridge this gap by presenting a comparative evaluation of exemplar tracking-by-detection approaches, with different detector and tracker choices, on relevant public datasets. Based on exemplar experimental results obtained, it goes beyond to present generalized insights and discussions that highlight the influence of detector and tracker choices on tracking performance

Owing to their pervasive use in the literature, relevance, and performance in MOT Challenge [8], we consider five different trackers (filtering strategies): A Decentralized Particle Filter (DPF), e.g., [1], Tracker Hierarchy [9], Reversible Jump Markov Chain Monte Carlo - Particle Filter (RJMCMC) [5], Simple Online Real-time Tracker (SORT) [10], and Markov Decision Processes (MDP) [11] tracking. The DPF and RJMCMC are selected as they are the most popular Monte Carlo approaches; Tracker Hierarchy and SORT, on the other hand, showcase a deterministic like approach with mixed closed-form stochastic and deterministic tracking strategy. MDP is unique as it tries to model lifetime of a target with Markov Decision Processes, on top of similar target dynamics and appearance considerations.

The aforementioned trackers are coupled with six selected detectors, namely: Histogram of Oriented Gradients (HOG) based detector denoted as HOG-SVM [12] , Deformable Part-based Methods (DPM) detector [13], Aggregate Channel Features (ACF) based detector [2], Locally Decorrelated Channel Features (LDCF) [14] that builds upon ACF, and two deep learning based detectors namely Region-based Convolutional Neural Networks (RCNN) [15], and Deep Convolutional Neural Networks (DeepPed) [16].

These detectors mark distinct detector superiority era onsets as published in 2005, 2010, 2014 (x2), and 2015 consecutively. Furthermore, our choice is motivated by the fact that LDCF, ACF and DPM are amongst the current best detectors based on hand-crafted features, and DeepPed and RCNN are the prominent ones amongst deep learned feature based approaches. HOG-SVM, though not currently the best itself, uses features that are constituents, in one way or another, of current state-of-the-art approaches and historically has been the *de facto* benchmark detector. Here, it is important to mention that we do not consider detectors that use background/foreground subtraction techniques, e.g., [17], have been recently dominated in the literature by the success in the others (those that employ learned people models). In short, the selected trackers and detectors are quite relevant and representative for the intended comparative evaluation.

The rest of this paper is organized as follows: Sec. II

A. A. Mekonnen email: alhayat.ali@gmail.com.

F. Lerasle is with CNRS, LAAS, 7, Avenue du Colonel Roche, F-31400 Toulouse, France; and Univ de Toulouse, UPS, LAAS, F-31400 Toulouse, France e-mail: lerasle@laas.fr.

presents related work and highlights our contributions. Sec. III details the adopted tracking-by-detection framework, and it is followed by Sec. IV and Sec. V which describe the chosen visual people detectors and multi-object tracking methodologies in detail, respectively. Sec. VI details the experimental settings and obtained results; Sec. VII provides a comprehensive discussion, and Sec. VIII presents concluding remarks.

II. Related Work and Contributions

Tracking-by-detection is overwhelmingly present in MOT, especially in video surveillance and other people tracking systems due to recent advances in detector performance [6]. It provides a strong framework upon which multi-target trackers that are able to recover from drift, target loss, occlusions or target confusion, can be built. Moreover, it has been the principal way of tracking especially in multi-person tracking [1]. The main advantage of this method is that it allows multi-object trackers to rely on the output of the detector in order to give birth, kill, or correct a track.

Tracking-by-detection approaches in the literature can be roughly categorized into two: a batch and an online approach. The batch approach utilizes global optimization over an entire video content to determine target trajectories from detections, e.g., [17]. The online approach, on the other hand, can either by a first order Makovian system that employs a recursive probabilistic approach to update and refine the trajectory of targets on a frame by frame basis, e.g., [18], [1], or a tracklet based system that first generates short tracklets by linking frame by frame detections, which are then globally associated to build longer tracks, e.g, [19]. As their names suggest, the batch approach renders itself to offline use while the latter is suitable and mainly used for online tracking applications. In this work we focus on first order Markovian trackers. As illustrated in Fig. 1, tracking-by-detection contains three main components: a detector, a tracker (filter), and a data association module. The tracker by itself further encapsulates target appearance model, target dynamic model, and state-space exploration technique.

The earliest account of coupling learned detectors in tracking-by-detection can be traced back to the works of Beymer and Konolige [20] (though at that time it was not called as such). In their work, they made use of a learned template based person detector with a Kalman Filter based tracker. Tracking-by-detection paradigm became popular only after significant improvements in people detection were demonstrated a couple of years later [21], [22]. Until recently, the majority of tracking-by-detection works have resorted to the HOG people detector [1], [18], [9]. Even currently, given the vast pool of detectors with varying performances, only a handful of them have been investigated for tracking-by-detection – HOG and ACF to be precise, e.g., [1], [8]. The trend is to pick a detector and directly use it – to the best of our knowledge there is no comparative work that shows the effect of a detector choice on tracker performance in different application contexts. This work addresses this gap by carrying out detector-tracker choice evaluations on public datasets to highlight the gains and losses incurred under different contexts. In the following consecutive paragraphs, related works in people detection and tracking are briefly discussed.

As highlighted, the literature on people detection is quite vast (please refer to [3], [23] for extensive surveys), but to briefly recap it, early success was achieved using rudimentary Haar like features inspired by Haar Wavelets that capture region intensity differences, but have limited descriptive power [24]. These were later significantly improved with the use of gradient based HOG features [12]. Building on HOG, several works have proliferated improving the state-of-the-art by combining HOG with other features (utilizing heterogeneous pool of features). Examples include: the HogLBP detector [25] that combined HOG features with Local Binary Patterns (LBP), and the MultiFTR detector [26] that combined HOG with Haar like and shape context features. The next significant improvement was achieved by the Deformable Parts-based Model (DPM) detector which uses slightly altered HOG features in a parts based detector configuration that explicitly looks for different automatically learned parts of a person (five to be exact) to detect a person [13]. Following this, further improvements have been made possible with the advent of Channel Features [27] and their derivatives. Channel features combined with soft-cascade boosting frameworks ([28], [29]) are amongst the current best approaches both in terms of detection performance and computation time [30].

MOT, another core component of tracking-by-detection, can be interpreted as the process of accurately estimating the state of targets – location, identity, and dynamic configuration – over time from a set of observations, e.g., [4]. Without loss of generality, MOT is considered to refer to tracking of multiple people henceforth. Two main paradigms exist for MOT state representation, which also indirectly govern the state-space exploration technique: A centralized approach, in which all the states of the tracked targets are joined, as subspaces, to yield a single representation that captures the entire configuration of the tracked persons [5], [31], [32]; and a decentralized approach whereby each target is represented, and consequently tracked independently, *e.g.*, [1]. The advantage of the joint representation is, should the targets interact, an interaction model can be incorporated in the tracking problem and tackled systematically. On the other hand, for the independent representation, target interaction models can not be incorporated directly. It naturally lends itself to ad-hoc solutions based on a higher level supervisor which manages the trackers' behaviors during close-by interactions [18], [1].

The tracker is further governed by the adopted target dynamic and appearance model. The target dynamic model dictates how the targets evolve in the current time frame from the previous state. In MOT, it is common to consider random walk, *e.g.*, [31], [33], linear autoregressive models with constant velocity, *e.g.*, [1], and non-linear models, for example, in the form of social forces [34], to describe target evolution. On the subject of appearance model, trackers can usually be classified in either of two groups: generative [35], [36] and discriminative methods [37], [38]. Generative models usually learn a model that directly represents the target object or person and use it to rate candidate positions, either by using

back-projection to find the region with minimal reconstruction error [9] or by sampling [39] to find the best candidate. Discriminative methods, on the other hand, prefer to model tracking as a binary classification task in order to discriminate the target from the background in the feature space [37], [38]. For these reasons, recent trackers have tried to make detectors and trackers work hand-in-hand in a framework that leverages their respective strengths. For example, [40], [1] complement the generic appearance model of its detector with online-trained classifiers that are able to accurately discriminate between targets.

***Contributions**:* The main goal of this work is a systematic evaluation of various detector-tracker combinations (that define a specific tracking-by-detection approach), on several public datasets to assess the effect of detector and tracker choices on performance under different applicative contexts. To achieve this, we consider a combination of six visual people detectors and five multi-object trackers, and evaluate their performance on seven public datasets. This is a very challenging endeavor and a crucial contribution to the state-of-the-art that is lacking this kind of experimental benchmark. Hence, our main contributions can be summarized as: (1) Systematic evaluation of tracking-by-detection approaches based on relevant combination of trackers and detectors on different application contexts; and (2) Based on the experimental results obtained using the chosen detectors and trackers, we try to go beyond and present generalized insights and discussions that highlight the influence of detector and tracker choices on tracking performance

III. Framework

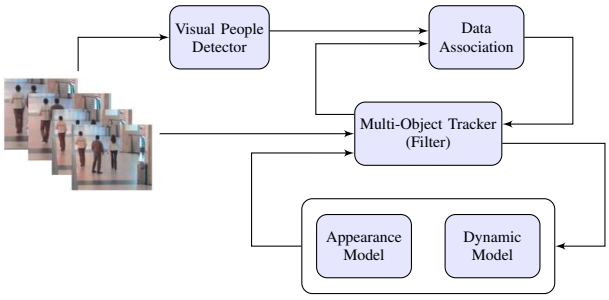


Fig. 1: Basic components of a tracking-by-detection framework and their interactions.

The adopted tracking-by-detection framework is depicted in Fig. 1. It builds upon object detectors to initialize, terminate, and update tracks. It is important to mention here that all tracking in this work is carried out on the image plane, with the output of a tracker describing a bounding rectangular box delineating the target. Detections and tracks are matched by a data association module that identifies detections as: one of the targets, in which case it is used to update the track, or as a new target, in which case it is used to create a potential track that awaits further associations with future detections to become a track. The tracker, or filter, itself will then handle how tracks are propagated at the current frame given a dynamic model, and estimate the most probable bounding box of the target at the current frame using an appearance model. The filter's purpose is to compensate for the detector's unreliability, i.e., high number of false positives and false negatives, and discrete number of responses in opposition to, for instance, a likelihood map.

The visual detector outputs a set of unlabeled detections, along with their *detector confidence*, from the current image. Detector confidence can be thresholded in order to reduce false detections. The list of detectors considered are presented in detail in Sec. IV. These detections are then sent to the data association module which matches detections in the current frame with tracks from previous time frames – if any – using the detections' and the tracks' location, scale, and appearance, via a matching score $S_{i,j}$ (for the ith detection and jth track) computed as:

$$S_{i,j} = \lambda_d \left\| \begin{matrix} \mathsf{x}_i - \mathsf{x}_j \\ \mathsf{y}_i - \mathsf{y}_j \end{matrix} \right\|_2 + \lambda_s \left\| s_i - s_j \right\|_2 + \lambda_{app} \mathcal{B}(\mathcal{H}_i, \mathcal{H}_j) \quad (1)$$

Where λ_d, λ_s and λ_{app} are adjustable parameters, and $\mathcal{B}(\mathcal{H}_i, \mathcal{H}_j)$ is the Bhattacharyya color histogram similarity coefficient. x, y denote position on the image plane, and s denotes scale. Scores are thresholded to ensure unlikely associations (i.e., associations with low scores) are not made. Even though, the Hungarian algorithm is usually used for the association using the score matrix, we use a greedy algorithm that iteratively associates detections and tracks with the highest score as it has been shown to be sufficient [1].

When a detection is assigned to a track, the track's remaining lifespan is increased, whereas tracks that are not matched to a detection see their lifespan decrease. When a track remains unmatched to any detection for a certain number of frames, the track is killed. The unmatched detections are used to create potential new tracks, which become an active track when associated with sufficient number of detection consecutive frames. Once data association is performed, the tracking (filtering) process takes place. The list of considered trackers are described in Sec. V.

IV. Visual People Detectors

This section presents the different visual people detectors investigated in this work. As presented in Sec. II, the state-of-the-art in visual people detector encompasses several detectors that have different detection performance, computation time, and model abstraction. In this work, we select six detectors, namely: HOG-SVM [12], DPM [13], ACF [2], LDCF [14], DeePeed [16], and RCNN [15] for the intended tracking-by-detection evaluations. Relevant characteristics of these detectors are summarized in Table I.

TABLE I: Summary of the six detectors investigated.

Detector	Feature Type	Model	Classifier	NMS
HOG-SVM [12]	HOG	Holistic	Linear SVM	PM
DPM [13]	HOG	Parts-based	Latent SVM	PM
ACF [2]	Channel Features	Holistic	AdaBoost	PM
LDCF [14]	Channel Features	Holistic	AdaBoost	PM
DeepPed [16]	Deep Learned (CNN)	Holistic	Linear SVM	PM
RCNN [15]	Deep Learned (CNN)	Holistic	Linear SVM	PM

A. Histogram of Oriented Gradients (HOG-SVM)

This detector, proposed by Dalal and Triggs [12], is one of the classical and oldest detectors. This detector computes local histograms of the gradient orientation on a dense grid and uses linear Support Vector Machine (SVM) as a classifier. The learned model is based on a holistic (full-body) abstraction trained on the public INRIA person training dataset [12]. The detection outputs are filtered by a Pairwise Max (PM) Non-Maximal Suppression (NMS) technique that suppresses the less confident of every pair of detections that overlap sufficiently. Although HOG-SVM is not the current best detector, its constituent HOG features are the most discriminant features to date [3].

B. Deformable Parts Model (DPM)

DPM [13] is a parts based detector that works by aggregating evidence of different parts of a body to detect a person in an image. The detector uses a mixture of deformable part-based models and a modified version of HOG features. The model consists of a root filter (one that characterizes full-body) and several part filters; its score over a candidate window is determined as the score of the root filter plus sum of the scores of each part filters, taking the maximum over placements of the parts, minus a deformation cost that penalizes deviation from ideal part locations relative to the root filter. It is learned using partially labeled data with a latent SVM, on the INRIA person dataset. The final detection bounding box is determined with a learned mapping function that uses the detected parts' positions. It uses a PM based NMS technique. Since this detector relies on parts, it detects partially occluded people well and leads to better localization accuracies.

C. Aggregate Channel Features (ACF)

This is a fast person detector based on the notion of channel features that has outperformed several detectors on various benchmarking datasets [2]. It is based on aggregates of features represented as channels, a variant of Boosted classifier, and a holistic person abstraction. A channel is a per-pixel feature map computed from a corresponding patches of input pixels. It can, for example, be the L component of the LUV color transformed input image, or even a histogram of each quantized gradient orientation (one channel per orientation) of the input image. ACF uses ten channels – gradient magnitude, HOG (6 channels), and LUV color channels. Each channel is aggregated over blocks to create lower resolution channels. The final classifier is learned using AdaBoost and depth two decision trees over these channel features. The detector considered in this work is trained on the INRIA person dataset and uses PM based NMS.

D. Locally Decorrelated Channel Features (LDCF)

The LDCF people detector [14] is a detector that also relies on channel features like ACF. But, instead of training a classifier on the features directly, it applies a decorrelation step beforehand. The key point is the observation that decision trees used in Boosting, that use orthogonal (single feature) splits, can generalize better if the correlation between channel features is reduced. Hence, LDCF modifies ACF by applying decorrelating filters per channel. The filters are determined as the eigen vectors of a channel specific covariance matrix computed from a large collection of natural images.

E. Deep Convolutional Neural Networks (DeepPed)

This people detector, referred as DeepPed [16], is a pedestrian detection system based on deep learning, adapting a general-purpose convolutional network to the task at hand. The detector employs a combination of LDCF as region proposal algorithm, a finetuned deep convolutional neural network for feature extraction, and a linear SVM for the final classification. Due to the complexity of the different stages and parameters involved, it is not possible to provide useful details here. But, the actual detector pipeline with the utilized parameter values is detailed in [16] dubbed as DeepPed.

F. Region-based Convolutional Neural Networks (RCNN)

The RCNN people detector is based on the works of Girshick et al. [15]. This detection system consists of three modules. The first generates category-independent region proposals. These proposals define the set of candidate detections available to the detector and are based on selective search. The second module is a large convolutional neural network that extracts a fixed-length feature vector from each region (each selected region is warped into a pre-defined and fixed rectangular region). The third module is a class specific linear SVM that classifies each fixed-length feature as a person or not. Even though the detector presented in [15] is applied to several classes, in this work, the model trained for pedestrian detection is utilized.

The selected six people detectors highlight advances made by different generation of detectors. Both HOG-SVM and DPM are based on similar underlying HOG features. LDCF and ACF are based on the notion of channel features. On the other hand, DeepPed and RCNN are based on deep learned features and represent recent advances made in machine learning. On recent benchmarks made on public datasets [41], [42], [16], [15], DeepPed and RCNN show better performance, followed by LDCF, ACF, and DPM respectively. HOG-SVM performs poorly, nevertheless, it is important to include it in benchmarking as it marks the first significant advance made by a people detector. Detection evaluation results on public datasets used in this work are presented in Section VI-C.

V. Multi-Object Trackers (MOT)

MOT is considered here in the vein of multi-person tracking utilizing a tracking-by-detection paradigm as illustrated in Sec. III. We consider two classes of trackers, purely probabilistic ones – based on sequential Monte Carlo approach (DPF and RJMCMC) and Markov Decision Processes (MDP) – and a deterministic like approach with mixed closed-form stochastic and deterministic tracking strategy (Tracker Hierarchy and SORT). Furthermore, the selected trackers can be categorized as decentralized (DPF, Tracker Hierarchy, SORT, and MDP), and centralized (RJMCMC). These trackers, combines with the six detectors, are evaluated on seven publicly available benchmarking datasets (Sec. VI). Table II summarizes important characteristics of the considered trackers; details are provided subsequently.

TABLE II: Summary of considered tracking (filtering) types. IS: importance sampling, MH: Metropolis-Hastings sampling, MMSE: minimum mean square error, KF: Kalman Filter.

Tracker	Sampling	Appearance Model	Dynamic Model	Data association	Point estimate
DPF	IS	Multi-Template	Random walk	Greedy	MMSE
RJMCMC	MH	Multi-Template	Random walk	Greedy	MMSE
Hierarchy [9]	–	Multi-Template	Linear velocity (KF)	Hungarian	Mode (Mean shift)
SORT [10]	–	–	Linear velocity (KF)	Hungarian	Mean
MDP [11]	–	Single-Template	Optical flow	Hungarian	Lucas-Kanade [43]

A. Decentralized Particle Filter (DPF)

In this approach, each target is assigned a unique instance of a Particle Filter as a tracker. This target specific tracker is based on the ICondensation [39] filter, a sequential Monte Carlo approach which approximates the posterior over the target state x_t given all measurements up to time t, $Z_{1:t}$, using a set of N weighted samples, i.e., $p(x_t|Z_{1:t}) \approx \{x_t^{(i)}, w_t^{(i)}\}_{i=1}^{N}$. Tracking is achieved sequentially with the notion of Importance Sampling whereby the particles at time $t-1$ are propagated according to a proposal density $q(.)$ and their weights are updated in accordance with Eq. 2. The DPF utilized here is the same as presented in [7] but with improved multi-template appearance model. The exact label for the complete tracker is derived by appending the detector name on tracker, e.g., DPF-ACF.

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{t-1}^{(i)}, z_t)} \tag{2}$$

a) State Space: The state of a target j at time t in particle i, $x_t^{(i)}$, is represented by the vector $[\mathsf{x}_t^i, \mathsf{y}_t^i, \mathsf{s}_t^i]^T$, where x and y denote the position, within the image plane, and s denotes the scale of the rigid boundary box encapsulating the target at time t. Here, showing the target index j is not relevant as a single DPF tracks only one target.

b) Particle Sampling: The importance function used to sample particles $q(x_t^{(i)}|x_{t-1}^{(i)}, z_t)$ is described in Eq. 3. This proposal distribution propagates β proportion of the particles from the previous time frame with the target dynamics model, sample α proportion of the particles from the detector proposal $\pi(x_t^{(i)}|z_t)$, and sample the remaining particles from the prior distribution (α, β, γ should sum to 1). This allows the tracker to incorporate detection information into the filtering step.

$$q(x_t^{(i)}|x_{t-1}^{(i)}, z_t) = \beta p(x_t^{(i)}|x_{t-1}^{(i)}) + \alpha\pi(x_t^{(i)}|z_t) + \gamma p_0(x_t^{(i)}) \tag{3}$$

The detector proposal distribution $\pi(x_t^{(i)}|z_t)$ is modeled as a Normal distribution $\mathcal{N}(x_t^{(i)}|z_t^{(i)}, \Sigma_d)$, where $z_t^{(i)}$ indicates a detected target that has been associated with particle i, and Σ_d is the covariance matrix of the detector $d \in \{$HOG-SVM, DPM, ACF, LDCF, DeepPed, RCNN$\}$.

c) Dynamic Model: Given the nature and variability of the evaluation datasets (Sec. VI-B) and difficulty of characterizing motion of humans on the image plane, a random walk dynamics model is used, i.e., for a particle i, $p(x_t^{(i)}|x_{t-1}^{(i)}) = \mathcal{N}(x_t^{(i)}|x_{t-1}^{(i)}, \Sigma_{dyn})$.

d) Appearance Model: The appearance of a tracked target j is kept track of using a dynamically updated multi-template (MT) histograms (template ensemble) similar to [9].

In this case, the appearance of tracked target is captured by the set of histograms $\left\{\mathcal{H}^j_{\mathsf{c}_k,k}\right\}_{k=1}^{N_T}$. Here, j identifies the target, c_k identifies the two color channels of the k^{th} histogram (see [9] for details). With this change, the likelihood measure of a DPF is given by Eq. 4.

$$p(z_t|x_t^{(i)})|_j \propto exp\left(-\lambda \sum_{k=1}^{N_T} \mathcal{B}\left(\mathcal{H}^j_{\mathsf{c}_k,k}, \mathcal{H}^{x_t^{(i)}}_{\mathsf{c}_k,t}\right)\right) \tag{4}$$

$\mathcal{H}^{x_t^{(i)}}_{\mathsf{c}_k,t}$ stands for the histograms (with color channels c_k) computed from the image frame at time t at the bounding box specified by $x_t^{(i)}$. The target template ensemble update is managed as in [9].

e) Tracker Birth and Death: The creation of a new track and the removal, or death, of an instantiated track is managed in an ad-hoc manner as in [1], [18]. Whenever there is unassociated detection, a new potential track, one instance of DPF, is created. Once N_{birth} number of consecutive detections have been associated with it, it is upgraded to a real track and marks the birth of a new tracked target. Similarly, when a tracked target has not been associated with a detection for N_{death} number of consecutive frames, it is removed and the target is no longer tracked.

B. Reversible Jump Markov Chain Monte Carlo - Particle Filter (RJMCMC)

RJMCMC, proposed in [44], [5], is a popular joint state (centralized) tracker that represents the state of all tracked targets with a single state vector. Unlike classical joint state particle filters that are based on Important Sampling, RJMCMC relies on the Metropolis-Hastings (MH) algorithm [45] for sample generation. Similar to DPF, RJMCMC approximates the posterior over the tracked targets' state X_{t-1} given all measurements up to time $t-1$, $Z_{1:t-1}$, using a set of M particles. But, this time the particles are *unweighted*, i.e., $p(X_{t-1}|Z_{1:t-1}) \approx \{X_{t-1}^{(i)}\}_{i=1}^{M}$. The posterior at the current time frame is approximated with Eq. 5. $X_t = \left\{x_t^j\right\}_{j=1}^{|X_t|}$ represents the states of all tracked targets. $|X_t|$ denotes the total number of tracked targets.

$$p\left(X_t|Z_{1:t}\right) \quad \propto p(z_t|X_t) \sum_{i=1}^{M} p(X_t|X_{t-1}^{(i)}) \tag{5}$$

RJMCMC defines a Markov Chain over the state configuration so that the stationary distribution of the chain approximates the posterior distribution in Eq. 5. It uses a set of m moves to change the dimension of the state, i.e., adding new target, removing untracked targets, or leave it unchanged according to a prior move proposal q_m. Each move m is associated with a move specific proposal distribution $Q_m(.)$, and must have a reverse move m^* that assures reversibility so that detailed balance will be achieved and the chain will converge to the desired stationary distribution [5]. During the estimation process, at the i^{th} iteration, it first samples a move from q_m and proposes a new particle X^* based on $Q_m(.)$. It then computes the acceptance ratio α_a (Eq. 6) with $Q_m(.)$ and the reverse move proposal distribution $Q_{m^*}(.)$. The proposed particle is accepted with probability α_a or otherwise rejected.

The *burn-in*, M_b, and *thin-out*, M_{th}, particles are discarded leaving M unweighted samples to represent the posterior.

$$\alpha_a = \min\left(1, \frac{p(X^*|Z_{1:t})Q_{m^*}(X_t^{(i-1)}; X^*)q_{m^*}\Psi(X^*)}{p(X_t^{(i-1)}|Z_{1:t})Q_m(X^*; X_t^{(i-1)})q_m\Psi(X_t^{(i-1)})}\right) \tag{6}$$

$\Psi(.)$ is the interaction model. The implementation in this work is similar as in [7] with the addition that the appearance model now is based on multi-template histogram ensemble.

a) State Space: The state vector of a person j in particle i at time t is a vector $x_{j,t}^i = [Id_{j,t}^i, \mathsf{x}_{j,t}^i, \mathsf{y}_{j,t}^i, \mathsf{s}_{j,t}^i]^T$, where $\mathsf{x}, \mathsf{y}, \mathsf{s}$ denote the position and scale of a target in the image plane, and Id indicates the identity of the target. Consequently, the i^{th} particle at time t is represented as $X_t^i = \{I_t^i, x_{(j,t)}^i\}, j \in \{1, ..., I_t^n\}$, where I_t^i is the number of tracked persons in the i^{th} particle.

b) Proposal Moves: The set of proposal moves considered are: $m = \{add,\ delete,\ remove,\ stay,\ update,\ swap\}$. The choice of the proposal privileged in each iteration is determined by q_m, the jump move distribution.

c) Interaction Model: As in [44], [5], we adopt a pairwise Markov Random Field (MRF) where the cliques are restricted to the pairs of nodes (targets define the nodes of the graph), that are directly connected to the graph. This is given by Eq. 7, where $\mathcal{D}(,)$ denotes the normalized Euclidean distance between the state vectors.

$$\Psi(X_t^i) = \Pi_{j \neq k}\left(1 - e^{-\lambda_{mrf}\mathcal{D}^2(x_{j,t}^i, x_{k,t}^i)}\right) \tag{7}$$

d) Appearance Model: The appearance model is based on multi-template histogram ensemble similar to the DPF. The difference is the way $p(z_t|X_t^{(i)})$ gets computed for the i^{th} particles since $X_t^{(i)}$ contains the states of all tracked targets. Let $\hat{X}$ denote the subset of updated or swapped targets (excluding removed or added targets whose likelihood is set to one). Then the likelihood for particle $X_t^{(i)}$ is computed via Eq. 8..

$$p(z_t|X_t^{(i)}) \propto exp\left(-\lambda \sum_{x \in \hat{X}} \mathcal{I}(j,x) \sum_{k=1}^{N_T} \mathcal{B}\left(\mathcal{H}_{\mathsf{c}_k,k}^j, \mathcal{H}_{\mathsf{c}_k,t}^x\right)\right) \tag{8}$$

$$\mathcal{I}(j,x) = \begin{cases} 1, & \text{if } j = id(x) \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

The indicator function $\mathcal{I}(j,x)$, Eq. 9, insures a target's appearance model is correctly matched with the correspond target in the state vector. The indicator function $\mathcal{I}(j,x)$ is defined as in Eq. 9.

The RJMCMC is coupled with the different detectors presented and evaluated. The coupled tracker-detector is denoted using RJMCMC followed by used detector acronym, e.g., RJMCMC-ACF. Similar to DPF, the detection-track data association is handled via a greedy assignment algorithm.

C. Tracker Hierarchy (Hierarchy)

This multi-object tracker is another tracking-by-detection decentralized MOT that assigns a single tracker per target. It is a tracker that consists of a rich appearance model of the target in the form of a template ensemble and uses hierarchy of expert and novice trackers for efficient multi-person tracking. It alternates between mean-shift mode estimator (to consider the target's appearance) and a Kalman filter (to account for the target's linear velocity motion dynamics) [9]. We consider evaluating its performance with different detectors as it, unlike the two other trackers, adopts a mixed deterministic and closed form stochastic estimation process. Please refer to [9] for further details. This tracker combined with any of the detectors is labeled as Hierarchy followed by detector name, e.g., Hierarchy-ACF.

D. Simple Online and Realtime Tracker (SORT)

The SORT tracker, proposed by Bewley et al. [10], is a lightweight multi-object tracker that depends solely on detections and dynamic motion model without using any target appearance model for tracking. The trackers focuses on efficient and reliable handling of the common frame-to-frame associations. Additionally, it employs two classical yet extremely efficient methods, Kalman filter and Hungarian method [46], to handle the motion prediction and data association components of the tracking problem respectively.

The tracker tracks each target independently and approximates the inter-frame displacements of each target with a linear constant velocity model which is independent of other objects and camera motion. In assigning detections to existing targets, each targets bounding box geometry is estimated by predicting its new location in the current frame. The assignment cost matrix is then computed as the intersection-over-union (IOU) distance between each detection and all predicted bounding boxes from the existing targets. The assignment is solved optimally using the Hungarian algorithm. Tracker birth and death are handled similar to the DPF presented in Sec. V-A. In our experiments, this tracker is evaluated by using each of the six presented detectors. Each variant is postfixed with the name of the associated detector, e.g., SORT-ACF.

E. Multi-object Tracking by Markov Decision Processes (MDP)

The MDP tracker [11], formulates the online MOT problem as decision making in Markov Decision Processes (MDPs), where the lifetime of an object is modeled with an MDP. Learning a similarity function for data association is equivalent to learning a policy for the MDP, and the policy learning is approached in a reinforcement learning fashion which benefits from both advantages of offline-learning and online-learning for data association. In this framework, a single object tracker is considered to be an agent in MDP, whose task is to track the target. Then good policies are learned for the MDP with reinforcement learning, and multiple MDPs are employed to track multiple targets.

Given a new input video frame, targets in tracked states are processed first to determine whether they should stay as tracked or transfer to lost states. Then a pairwise similarity between lost targets and object detections which are not covered by the tracked targets is computed. After that, the similarity scores are used in the Hungarian algorithm to obtain the assignment between detections and lost targets. According to the assignment, lost targets which are linked to some object detections are transferred to tracked states. Otherwise, they

stay as lost. Finally, an MDP is initialized for each object detection which is not covered by any tracked target. To propagate the tracked targets to the next frame (target dynamic model), optical flow from densely and uniformly sampled points inside the target window is used. The corresponding point estimate location in the new frame is determined using the iterative Lucas-Kanade [43] method with pyramids. Additionally, the framework can naturally handle the birth/death and appearance/disappearance of targets by treating them as state transitions in the MDP while leveraging existing online single object tracking methods. Please refer to [11] for further details. Similar to the other trackers, this tracker is evaluated by using each of the six presented detectors. Each variant is postfixed with the name of the associated detector, e.g., MDP-ACF.

VI. Experiments and Results

This section presents the different experiments carried out in detail. The objective is to evaluate the different tracking-by-detection approaches on public datasets in order to gather useful insights on the detector-tracker choice and more specifically on how each approach behaves under varying applicative contexts. We first evaluate the detectors' performances on each dataset and then proceed with tracking-by-detection evaluation. The section begins with a presentation of the evaluation metrics and the datasets used, and follows with a description of the specific experimental/implementation settings used and obtained results.

A. Evaluation metrics

Detector performance is measured in terms of *precision* and *recall* which are defined according to Eq. 10. Precision characterizes the proportion of detections that are indeed true targets, whereas recall indicates the proportion of correctly detected targets. TP stands for true positives, FP for false positives, and FN for false negatives. These values are determined based on the per image evaluation described in [3].

$$Precision = \frac{TP}{TP+FP} \qquad Recall = \frac{TP}{TP+FN} \tag{10}$$

Tracker performance, on the other hand, is quantified using the prevalent CLEAR-MOT metrics [47]. The CLEAR-MOT metrics are principally based on computation of two quantities: the multi-object tracking accuracy (MOTA) and the multi-object tracking precision (MOTP), Eq. 11.

$$\text{MOTA} = 1-(\mathcal{F}_{\mathcal{P}}+\mathcal{F}_{\mathcal{N}}+\mathcal{I}d_{sw}); \quad \text{MOTP} = \frac{\sum_{j,t}\mathcal{S}(x_j,g_t)}{\sum_t c_t} \tag{11}$$

Where $\mathcal{F}_{\mathcal{P}} = \sum_t \frac{FP_t}{g_t}$ denotes the total number of false positive, $\mathcal{F}_{\mathcal{N}} = \sum_t \frac{FN_t}{g_t}$ denotes the total number of false negatives, and $\mathcal{I}d_{sw} = \sum_t \frac{Id_{sw,t}}{g_t}$ denotes the total number of id switches, divided by the total number of ground truth targets (g_t) summed over the entire dataset. Even though it is not directly computed, the true tracking rate (true positive) can be expressed as $\mathcal{T}_{\mathcal{P}} = \sum_t \frac{TP_t}{g_t}$. MOTP is the average bounding box overlap (intersection over union) between the estimated target position and ground truth annotations over the correctly tracked targets. A tracker estimated rectangular position $\mathcal{R}(x_j)$ is considered a correct track if its overlapping area score $\mathcal{S}(x_j,g_t) = \frac{\mathcal{R}(x_j)\cap\mathcal{R}(g_t)}{\mathcal{R}(x_j)\cup\mathcal{R}(g_t)}$ with the ground truth annotation g_t is above a threshold $\mathcal{S}_o$.

B. Datasets

For evaluation, we use seven publicly available datasets summarized in Tables III and IV. These datasets are selected so as to encompass varying target characteristics, environment contexts, and sensor configurations. They include: static/mobile camera, differing image frame resolutions, indoor/outdoor settings, cluttered/uncluttered backgrounds, repeated target occlusions, and several target interactions. As can be seen (observed) from Fig. 2 (Tables III and IV), PETS-S2L1 features an outdoor scene captured using a surveillance camera with a slanted perspective view; it has several occlusions and inter-target interactions. The CAVIAR-OneShop dataset features intermittent target occlusions in an indoor scenario and the targets' speeds vary, with some of them remaining static for some time. Similarly, CAVIAR-EnterExit features the same environment as CAVIAR-OneShop with more diverse directions of movements and several background clutters. TUD-Crossing features pedestrians crossing the road from a side view (static camera) with bi-directional horizontal target motions in a dense crowd. It has the most severe inter-target occlusions. The ETH-Bhnhof, ETH-Jelmoli, and ETH-Sunnyday datasets are all acquired using a moving camera. In ETH-Bahnhof the camera is mounted at hip-height and most targets walk towards or away from the camera on a crosswalk. In ETH-Jelmoli, the camera shows an erratic movement amidst a crowd of people moving in different directions on a plaza. The scene gets very complex though the crowd remains sparse. In ETH-Sunnyday the camera is moving forward on a crosswalk in a dense crowd. Targets move towards and away from the camera. Fig. 2 shows sample random frames taken from each datasets.

The six detectors are evaluated on all the presented datasets using *precision-recall* metrics. Fig. 3 shows the precision-recall curves generated by varying the final detection threshold. Generally speaking, all the detectors except HOG-SVM perform well on PETS-S2L1, TUD-Crossing, and ETH-Sunnyday, achieving a higher than 80% recall at some operating points. But, on the rest perform moderately with severe under-performance observed in CAVIAR-OneShop and ETH-Jelmoli. The operating point of each detector on these datasets is determined by *globally* setting the detector output threshold value to a point that maximizes the F1-score $\left(= 2.\frac{precison.recall}{precision+recall}\right)$. This point determines an operating point that balances precision and recall trade-offs equally. Accordingly, the thresholds for each detector are denoted as θ_d where $d \in \{\text{RCNN, DeepPed, LDCF, ACF, DPM, HOG-SVM}\}$ (see values in Table A.1 of [48]). The exact *precision-recall* obtained for all the datasets using these thresholds are shown in Table III. This table helps highlight each detector's performance on the different datasets. It will later be used as a basis to compare performance with detector alone and with tracker incorporated. LDCF records the highest recall and precision in three and six of these datasets respectively. RCNN and DPM demonstrate the highest recall in the remaining four datasets. HOG-SVM

(a) CAVIAREnterExit (b) CAVIAROneShop (c) PETS-S2L1 (d) TUD-Crossing (e) ETH-Bahnhof (f) ETH-Sunnyday (g) ETH-Jelmoli

Fig. 2: Sample images taken from each dataset.

and DeepPed show the worst recall rates. All in all, CAVIAR-OneShop and ETH-Jelmoli prove to be very difficult leading to very low recall rates (57% and 56% best case scenarios respectively).

C. Implementation details

Several implementation choices and experimental setups are discussed below. To tune the different free parameters related to the detectors and trackers (e.g., detector thresholds, number of particles, etc), a tuning dataset is used. For each evaluation dataset, a separate dataset acquired in the same setting is used for tuning. For both CAVIAR datasets, the CAVIAR-WalkByShop [49] is used for tuning; for PETS-S2L1, the corresponding PETS-S1L1 [50] dataset is used. The parameters used for all the three ETH datasets are tuned based on the ETH-Crossing [52] dataset. Finally, for TUD-Crossing, TUD-Campus [51] is used. The parameters for SORT and Hierarchy are tuned and that of MDP trained, in accordance with the approach outlined in their original publications using the corresponding tuning datasets (please refer to [10], [9], [11] for details). For DPF and RJMCMC, the tuning strategy is discussed below. The tuned parameters based on the PETS-S1L1 dataset for DPF and RJMCMC are shown in Table A.1 of the supplemental material [48]. Every experiment, tuning, and final evaluation, related to DPF and RJMCMC, is always averaged over ten runs to account for the stochastic nature of these filters and obtain meaningful statistics.

1) Detectors: The detectors used in the experiments are based on publicly available open source implementations: RCNN based on Girshick et al. [15] Python implementation; DeepPed based on Tome et al. [16] Matlab implementation; LDCF and ACF, based on Dollar's Matlab toolbox [53]; DPM based on the Matlab implementation released by Girshick et al. [54]; and the HOG-SVM detector based on OpenCV [55], the open source computer vision library, with slight modifications to provide continuous detection scores (the original provided only binary output).

The covariance matrix associated with each detector output is determined using the tuning datasets. The output of each detector, position and scale, is compared with the ground truth to define an error term for each detection-ground truth pair. Then the standard deviations of these samples are determined to compose the covariance matrices Σ_{d} (with diagonal components only) of the detector specific proposal distributions.

2) Trackers: The two particle based trackers, DPF and RJMCMC, are our implementations (in `C++`). For the Tracker Hierarchy, we use the original `C++` implementation from [9]. For SORT [10] and MDP [11], the `Python` and `Matlab` implementations provided by the authors, respectively, are used. The parameters of target dynamic model, random walk covariance matrix Σ_{dyn}, for DPF and RJMCMC are determined using the tuning datasets. Displacements of the targets in x, y, and s between consecutive frames are cached using the ground truth annotations. The variance of each variable is determined from this data and set as the diagonal elements of the covariance matrix. Sample values obtained based on PETS-S1L1 are listed in Table A.1 in [48]. There are more pronounced horizontal target displacements than vertical.

a) DPF: The sampling proposal distribution weights, Bhattacharyya coefficient scale, color histogram update rate, maximum number of templates, and track birth and death controllers are set to the values indicated in Table A.1 [48] based on observation from several works in the literature [9], [1], [56]. To determine the number of particles N to use, several runs are performed, on the tuning dataset, varying the number of particles and detector used. Then N is set to value that gives slightly better accuracy averaged across the different detectors used.

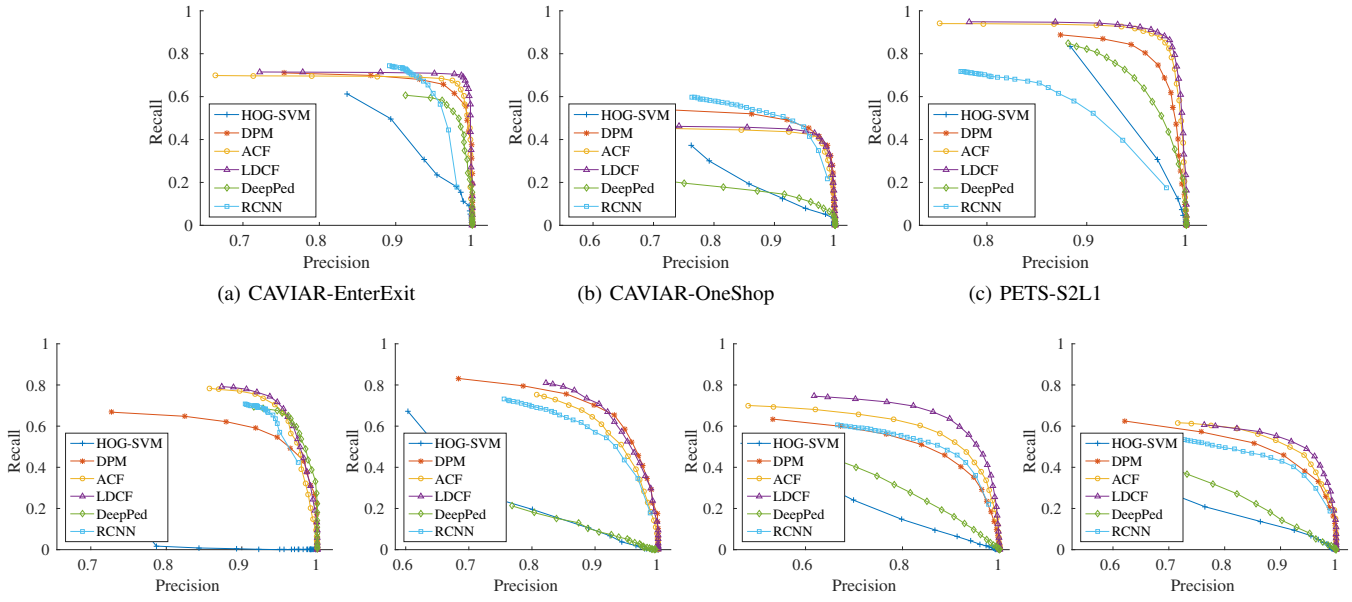
b) RJMCMC: For the RJMCMC tracker variants, the two most important parameters to tune are the number of effective particles M to use and the move proposal distribution q_m. The other parameters – dynamics of the targets, detector covariance matrix, and appearance model related parameters – remain the same as in DPF. To determine M, the tuning dataset is evaluated varying the number of particles used. This is done for each detector type. M is then set to a value that gives the maximum average MOTA.

Determining good values to use for RJMCMC's move proposal distribution q_m is very important and at the same time difficult. q_m is a vector composed of six continuous values corresponding to each move considered (add, delete, remove, stay, update, and swap). All these values should sum to one. Since exhaustive search is infeasible (continuous parameters and infinite possible combinations), we select a total of 14 intuitively selected combinations and construct the set $\{q_{m,k}\}_{k=1}^{14}$ (after preliminary visual inspection). If we denote the average MOTA obtained when using $q_{m,k}$ as $\mathrm{MOTA}_{q_{m,k}}$, then, the move proposal distribution to use is selected as the one that has the maximum MOTA score:

$$q_m = \underset{q_{m,k}}{\operatorname{argmax}} \left\{ \mathrm{MOTA}_{q_{m,k}} \right\} \tag{12}$$

Evidently, the MOTP does not vary a lot and stays more or less constant over a detector. For the PETS-S1L1 tuning, the actual values of the selected move proposal distributions are shown in Table A.1 [48].

3) Computation Time: The main objective of this paper is a systematic tracking-by-detection accuracy and precision performance evaluation to highlight detector and tracker choice trade-offs. Depending on the choice of detector and tracker,



(a) CAVIAR-EnterExit (b) CAVIAR-OneShop (c) PETS-S2L1

(d) TUD-Crossing (e) ETH-Sunnyday (f) ETH-Bahnhoff (g) ETH-Jelmoli

Fig. 3: Detector performance evaluation, in terms of precision-recall metrics, on all the public datasets.

TABLE III: Utilized public datasets with detector performance.

Dataset	Camera	Resolution	fps	#Frames	#Ids	Detector Recall / Precision						
						RCNN	DeepPed	LDCF	ACF	DPM	HOG-SVM	mean
CAVIAR-EnterExit [49]	static	384 × 288	25	383	4	**.73**/.91	.60/.93	.71/**.97**	.69/.94	.69/.89	.62/.84	.67/.91
CAVIAR-OneShop [49]	static	384 × 288	25	1377	6	**.57**/.82	.21/.73	.45/.93	.43/**.95**	.51/.88	.37/.76	.42/.85
PETS-S2L1 [50]	static	768 × 576	7	795	19	.70/.79	.85/.88	**.92**/**.95**	**.92**/.94	.85/**.95**	.81/.89	.84/.90
TUD-Crossing [51]	static	640 × 480	25	200	13	.70/.92	.69/.92	**.75**/**.93**	.74/**.93**	.63/.87	.53/.68	.67/.88
ETH-Bahnhof [52]	mobile	640 × 480	14	1000	224	.58/.75	.49/.57	**.70**/**.83**	.63/.78	.57/.73	.52/.47	.58/.69
ETH-Sunnyday [52]	mobile	640 × 480	14	354	30	.69/.81	.19/.79	.70/**.91**	.62/**.91**	**.76**/.85	.67/.60	.61/.81
ETH-Jelmoli [52]	mobile	640 × 480	14	440	74	.52/.76	.39/.71	.54/**.90**	.52/.87	**.56**/.79	.43/.52	.49/.76

TABLE IV: Characteristics of the utilized datasets.

Dataset	Environment	Camera View	Illumination Variation	Background Clutter	Occlusion	Interactions	Target Height (pixel)		
							min	max	average
CAVIAR-EnterExit [49]	indoor (corridor)	Perspective	⋆⋆	⋆	⋆	⋆	8.0	144.0	68.5
CAVIAR-OneShop [49]	indoor (corridor)	Perspective	⋆⋆	⋆	⋆⋆	⋆⋆	3.0	151.0	82.0
PETS-S2L1 [50]	outdoor (neighbourhood block)	Perspective	⋆	⋆	⋆⋆⋆	⋆⋆⋆	52.0	153.0	83.3
TUD-Crossing [51]	outdoor (road)	Fronto-parallel	⋆	⋆	⋆⋆⋆⋆	⋆	130.0	207.0	162.4
ETH-Bahnhoff [52]	outdoor (walkway)	Fronto-parallel	⋆⋆	⋆⋆	⋆⋆⋆	⋆⋆⋆	25.0	477.0	97.4
ETH-Sunnyday [52]	outdoor (walkway)	Fronto-parallel	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆	⋆⋆	48.0	468.0	135.2
ETH-Jelmoli [52]	outdoor (walkway)	Fronto-parallel	⋆⋆	⋆	⋆⋆⋆	⋆	38.0	459.0	116.1

the frame rate of the complete tracking-by-detection algorithm varies. As the detectors and trackers selected for evaluation are implemented in heterogeneous languages, e.g, Python, Matlab, and C++, and heterogeneous processors, CPU and GPU, it is impossible to determine a fair comparison. Nevertheless, we provide observed average frame rates during experimental evaluation here as a guide: For the detectors, HOG-SVM (∼3.8 fps), DPM (∼0.48 fps), ACF (∼23.4 fps), LDCF (∼5.4 fps), DeepPed (∼2 fps), and RCNN (∼4.7 fps); and for the trackers, DPF (∼8.2 fps), RJMCMC (∼1 fps), SORT (∼220 fps), Hierarchy (∼7.5 fps), and MDP (∼1.1 fps). These frame rates are obtained on ETH-Bahnhof dataset. Since the detection and tracking computations are decoupled, the overall detection-by-tracking algorithm frame rate reflects the contributions of both. For example, SORT-LDCF has ∼5.2 fps, and RJMCMC-DPM has ∼0.3 fps. This results are obtained on a PC with Intel Core i7-2720QM CPU, 8 GB of RAM, and NVIDIA Quadro 1000M GPU.

D. Results

Each detector-tracker combination is evaluated on the seven described public datasets using the parameter settings described in the previous section. Evaluation results are reported based on the MOTA and MOTP metrics. More importantly an overlap threshold $s_o = 0.5$ is used, according to established evaluation protocol [8], [57]. The raw outputs of each detector-tracker combination run on each dataset is available at [58]. Here, several summarized tables that highlight specific attributes are presented. They are categorized to make specific comparisons easy. The categories try to answer the following questions:

Q_1- Which tracking-by-detection combination does better?
Q_2- Which tracker, irrespective of utilized detector, performs better?
Q_3- Which detector, irrespective of utilized tracker, performs better?
Q_4- How do the different tracking-by-detection methods per-

form on mobile and static camera datasets (different contexts)?

1) Tracking-by-Detection Overall Assessment (Q_1): To determine the overall performance of each tracking-by-detection approach (detector-tracker combinations), results of experimental runs averaged over each dataset are presented. Tables V and VI present the average MOTA and MOTP results. In four out of the seven datasets, the SORT tracker shows the best tracking accuracy – in three combined with LDCF (SORT-LDCF) and in one combined with RCNN (SORT-RCNN). Similarly, MDP records the best tracking accuracy in remaining three datasets, combined with LDCF in two and with RCNN on the third one. With the exception of Hierarchy-RCNN, that achieves the second best accuracy on one dataset, the rest of the best and second best accuracy results are obtained with SORT and MDP trackers. Overall, on average, SORT-LDCF shows the best tracking accuracy of 62.8% across all the datasets followed by MDP-RCNN with 59.6%. A very important observation to make is that the best and worst average tracking accuracies across all datasets are obtained when combining LDCF with SORT and Hierarchy trackers, respectively. Compared to the rest, Hierarchy is much more sensitive to the detector choice.

Looking at average tracker precision, Table VI, the SORT tracker exhibits the best tracking precision on five of the seven datasets. SORT-ACF results in the best average precision of 77.8% across all datasets. This is seconded by SORT-DPM at 77.4%. The worst precision is obtained when using RJMCMC-HOG-SVM at 58.1%.

2) Tracker Assessment (Q_2): To analyze which tracker irrespective of used detector performs better, Table VII presents MOTA and MOTP results of each tracker family, averaged across all detectors per dataset. The results indicate that the MDP tracker achieves a better average MOTA in four of the datasets. In the other three, SORT gives the best average result. Overall, when averaged across the different datasets, MDP achieves a 47.4% average MOTA followed by SORT with 46.9%. Similarly, in precision, MDP leads to better results in five of the datasets followed by SORT in the remaining three. Irrespective of the detector choice, MDP manifests better filtering capabilities (higher $\mathcal{T}_\mathcal{P}$ and lower $\mathcal{F}_\mathcal{P}$) and better target localization. This is observed in the overall average results too. In terms of average tracking accuracy, the trackers can be ranked as follows: MDP, SORT, DPF, RJMCMC, and Hierarchy.

3) Detector Assessment (Q_3): Table VIII presents average tracking accuracies per detector (averaged across the five trackers). For ease of comparison, each detector's Recall/Precision (R/P) is also shown for each dataset. In three of the seven datasets, LDCF leads to best tracking accuracy results. DPM achieves best results on two datasets, and RCNN and ACF on single datasets each. Even though LDCF has the highest recall on four of the datasets, it is edged out by DPM which has the highest recall only on two datasets. When averaged across all datasets, DPM achieves an average MOTA of 42.1%. It is seconded by ACF which achieves 37.0%; LDCF ranks third with 36.8%. The worst result is demonstrated by HOG-SVM with an average MOTA of 18.0%. In terms of repeatability, DPM results in the lowest MOTA standard deviation across the different datasets and trackers. Based on average MOTA, detector ranking follows DPM, ACF, LDCF, RCNN, DeepPed, and HOG-SVM.

4) Performance on Mobile vs Static Camera Datasets (Q_4): Table IX presents MOTA and MOTP results averaged over the static (fixed) camera and mobile camera based datasets. The best tracking accuracy on static and mobile camera datasets are achieved by MDP-RCNN and SORT-LDCF, respectively. On the other hand, the second best results are obtained by SORT-LDCF and MDP-LDCF on static and mobile datasets, respectively. On average, the best static camera tracking accuracy is 7.8% higher than the mobile one. The best tracking precision on static datasets is obtained by both SORT and MDP combined with the DPM detector. On mobile datasets, it is achieved by SORT-LDCF. The best and worst detector choices for DPF, RJMCMC, Hierarchy, SORT, and MDP result in tracking accuracy margins of 12.5%, 38.0%, 58.1%, 39.2%, and 29.3% on static datasets, respectively. This becomes 33.1%, 62.4%, 70.3%, 41.9%, and 29.6% on mobile datasets, respectively. These are steep differences even though the highest difference in detector recall and precision is $< 15\%$.

VII. Discussions and Guidelines

We stir the discussion based on the questions, Q_1 - Q_4, raised in Sec. VI-D. The results presented in Tables V – IX provide very rich insights into the performance of the different detector-tracker combinations on different datasets/contexts. Undoubtedly, the performance of each tracking-by-detection approach is significantly influenced by the detector and tracker choices.

Consider the overall performance of tracking-by-detection approaches on each dataset (i.e., Q_1 and results in Tables V and VI). On the two CAVIAR datasets (EnterExit and OneShop), there is significant average tracker accuracy difference because of detector choice This is with a maximum of six targets and slight illumination variation due to shadows of the indoor vertical columns. The overall performance on CAVIAR-OneShop is very low due to the low detector recall and precision. In these two datasets, since the background environment does not change, any false positive occurrence is likely to recur exacerbating the overall performance. The PETS-S2L1, on the other hand, is an easier dataset even though there are several occlusions and target interactions. The background clutter is minimal and as a result of the mount position of the camera, the corresponding target displacement on the image plane is small. Consequently, the maximum tracking accuracy is reported on this dataset.The fourth dataset, TUD-Crossing, exhibits a significant inter-target occlusion (targets crossing on a zebra cross). MDP and SORT based trackers obtain the best tracking accuracies while Hierarchy based trackers do the worst. The three ETH datasets – ETH-Bahnhof, ETH-Sunnyday, ETH-Jelmoli – all have similar characteristics. On these datasets, SORT-LDCF and MDP-LDCF result in the best tracking accuracies. From the obtained results, it can be argued that no one tracking-by-detection approach exhibits the best performance on all datasets. Depending on the nature of

TABLE V: Tracking-by-detection average accuracy (MOTA↑ μ/σ) on the public datasets. The **best** and second best results, on each dataset, are highlighted.

Dataset	RCNN					DeepPed					LDCF				
	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP
CAVIAR-EnterExit	.773/.006	.468/.000	.798/.000	.767/.000	**.815**/.000	.488/.004	.476/.000	.621/.000	.541/.000	.582/.000	.550/.009	.623/.000	-.632/.000	.679/.000	.392/.000
CAVIAR-OneShop	.477/.006	.098/.016	.046/.000	**.516**/.000	.492/.000	.073/.002	.106/.011	.067/.000	.122/.000	.091/.000	.276/.004	.378/.000	-.410/.000	.416/.000	.034/.000
PETS-S2L1	.338/.002	-.095/.000	.511/.000	.617/.000	.685/.000	.581/.000	-.033/.000	.626/.000	.689/.000	.785/.000	.677/.000	.524/.033	.650/.000	**.836**/.000	.680/.000
TUD-Crossing	.492/.012	.022/.022	.054/.000	.664/.000	.698/.000	.439/.007	.400/.013	.137/.000	.588/.000	.637/.000	.338/.004	.486/.000	-.360/.000	.679/.000	**.763**/.000
ETH-Bahnhof	.285/.015	-.275/.000	.178/.000	.491/.000	.439/.000	.144/.011	.003/.000	.048/.000	.253/.000	.235/.000	.427/.012	.276/.000	-.180/.000	**.622**/.000	.381/.000
ETH-Jelmoli	.298/.008	-.339/.000	-.100/.000	.418/.000	.424/.000	.169/.007	-.050/.000	-.077/.000	.186/.000	.302/.000	.372/.004	.293/.000	-.109/.000	**.478**/.000	.454/.000
ETH-Sunnyday	.435/.004	-.254/.000	.147/.000	.574/.000	.616/.000	.087/.007	.026/.000	.111/.000	.088/.000	.238/.000	.591/.006	.434/.000	-.244/.000	.686/.000	**.828**/.000
Average	.443/.158	-.054/.266	.215/.300	.578/.109	.596/.139	.283/.199	.133/.203	.238/.258	.352/.231	.410/.241	.462/.138	.431/.119	-.184/.381	**.628**/.132	.505/.258

Dataset	ACF					DPM					HOG-SVM				
	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP
CAVIAR-EnterExit	.642/.011	.542/.000	.366/.000	.614/.000	.216/.000	.577/.004	.514/.015	.614/.000	.621/.000	.615/.000	.527/.005	.432/.049	.610/.000	.570/.000	.503/.000
CAVIAR-OneShop	.370/.002	.375/.000	-.152/.000	.260/.000	-.053/.000	.418/.002	.419/.000	.123/.000	.442/.000	.392/.000	.324/.003	.171/.000	-.355/.000	.323/.000	.271/.000
PETS-S2L1	.684/.000	.466/.000	.589/.000	.807/.000	.609/.000	.585/.009	.424/.000	.576/.000	.697/.000	.824/.000	.555/.011	.194/.000	.419/.000	.666/.000	.797/.000
TUD-Crossing	.338/.004	.456/.000	.203/.000	.469/.000	.730/.000	.379/.008	.381/.000	.246/.000	.420/.000	.527/.000	.256/.012	-.185/.000	.071/.000	-.064/.000	.359/.000
ETH-Bahnhof	.409/.005	.243/.000	-.075/.000	.465/.000	.203/.000	.317/.000	.204/.027	.076/.000	.403/.000	.291/.000	.129/.012	-.007/.027	-.653/.000	.221/.000	.201/.000
ETH-Jelmoli	.254/.010	.304/.000	.012/.000	.235/.000	.421/.000	.266/.006	.159/.025	.045/.000	.345/.000	.307/.000	.136/.010	-.398/.024	-.352/.000	.234/.000	.194/.000
ETH-Sunnyday	.318/.017	.373/.000	.146/.000	.229/.000	.781/.000	.539/.007	.329/.015	.146/.000	.591/.000	.731/.000	.344/.010	-.152/.077	-.682/.000	.212/.000	.423/.000
Average	.460/.149	.394/.117	.156/.243	.440/.205	.415/.291	.444/.149	.347/.120	.285/.220	.503/.124	.527/.195	.324/.159	.008/.264	-.135/.480	.309/.229	.396/.198

TABLE VI: Tracking-by-detection average precision (MOTP↑ μ/σ) on the public datasets. The **best** and second best results are highlighted.

Dataset	RCNN					DeepPed					LDCF				
	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP
CAVIAR-EnterExit	.750/.002	.674/.004	.744/.000	.780/.000	.777/.000	.679/.002	.680/.002	.684/.000	.658/.000	.659/.000	.760/.003	.728/.000	.571/.000	.778/.000	.786/.000
CAVIAR-OneShop	.733/.002	.615/.001	.695/.000	**.770**/.000	.760/.000	.684/.002	.653/.002	.674/.000	.664/.000	.669/.000	.718/.001	.654/.000	.576/.000	.727/.000	.728/.000
PETS-S2L1	.664/.001	.622/.003	.670/.000	.678/.000	.683/.000	.694/.001	.633/.004	.680/.000	.698/.000	.702/.000	.728/.000	.676/.000	.705/.000	.748/.000	.752/.000
TUD-Crossing	.721/.002	.604/.004	.704/.000	.746/.000	.753/.000	.679/.002	.636/.003	.660/.000	.672/.000	.680/.000	.664/.001	.696/.000	.558/.000	.766/.000	.770/.000
ETH-Bahnhof	.724/.000	.607/.000	.695/.000	.751/.000	.745/.000	.726/.000	.646/.000	.726/.000	.754/.000	.754/.000	.753/.000	.652/.000	.591/.000	.777/.000	**.780**/.000
ETH-Jelmoli	.717/.001	.599/.000	.655/.000	.765/.000	.742/.000	.729/.000	.629/.000	.700/.000	.747/.000	.773/.000	.756/.000	.638/.000	.606/.000	.780/.000	.777/.000
ETH-Sunnyday	.723/.002	.606/.006	.708/.000	.747/.000	.743/.000	.748/.004	.663/.002	.733/.000	.784/.000	.781/.000	.787/.001	.709/.000	.572/.000	.805/.000	.809/.000
Average	.719/.025	.618/.025	.696/.027	.746/.031	.743/.028	.705/.027	.648/.017	.694/.026	.714/.050	.717/.050	.738/.034	.679/.031	.597/.047	.768/.024	.772/.024

Dataset	ACF					DPM					HOG-SVM				
	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP	DPF	RJMCMC	Hierarchy	SORT	MDP
CAVIAR-EnterExit	.758/.001	.737/.000	.755/.000	.790/.000	.790/.000	.764/.003	.694/.000	.780/.000	.792/.000	**.801**/.000	.758/.003	.702/.000	.730/.000	.780/.000	.769/.000
CAVIAR-OneShop	.720/.000	.646/.000	.679/.000	.733/.000	.731/.000	.743/.001	.638/.000	.684/.000	.764/.000	.756/.000	.721/.001	.571/.000	.686/.000	.743/.000	.736/.000
PETS-S2L1	.729/.001	.680/.000	.702/.000	.755/.000	.753/.000	.728/.001	.654/.000	.707/.000	.756/.000	.757/.000	.735/.002	.583/.000	.689/.000	.756/.000	**.762**/.000
TUD-Crossing	.664/.001	.685/.000	.779/.000	**.782**/.000	.773/.000	.727/.003	.646/.000	.752/.000	.750/.000	.748/.000	.714/.004	.572/.000	.716/.000	.557/.000	.719/.000
ETH-Bahnhof	.751/.000	.637/.000	.691/.000	**.780**/.000	.766/.000	.744/.001	.622/.002	.706/.000	.779/.000	.775/.000	.696/.002	.558/.005	.643/.000	.711/.000	.719/.000
ETH-Jelmoli	.762/.000	.643/.000	.694/.000	**.791**/.000	.776/.000	.737/.003	.606/.001	.699/.000	.775/.000	.775/.000	.675/.002	.558/.002	.629/.000	.686/.000	.698/.000
ETH-Sunnyday	.781/.003	.706/.000	.721/.000	**.813**/.000	.805/.000	.768/.001	.651/.000	.752/.000	.802/.000	.793/.000	.720/.000	.522/.000	.632/.000	.706/.000	.732/.000
Average	.734/.038	.676/.035	.726/.035	**.778**/.024	.771/.024	.727/.032	.646/.026	.675/.033	.774/.018	.772/.019	.715/.025	.581/.054	.717/.039	.706/.069	.734/.024

TABLE VII: Tracker performance comparison, reported as μ/σ, on each dataset averaged across the different detectors. The **best** and second best MOTP and MOTA results are indicated.

Dataset	**DPF**		**RJMCMC**		**Hierarchy**		**SORT**		**MDP**	
	MOTP↑	**MOTA↑**	**MOTP↑**	**MOTA↑**	**MOTP↑**	**MOTA↑**	**MOTP↑**	**MOTA↑**	**MOTP↑**	**MOTA↑**
CAVIAR-EnterExit	.745/.030	.593/.118	.703/.024	.509/.195	.711/.070	.396/.276	.763/.048	**.632**/.141	**.764**/.049	.521/.092
CAVIAR-OneShop	.720/.019	.323/.080	.630/.030	.258/.288	.666/.041	-.114/.130	**.734**/.035	**.347**/.108	.730/.030	.205/.093
PETS-S2L1	.713/.026	.570/.168	.641/.034	.247/.274	.692/.014	.564/.338	.732/.032	.719/.231	**.735**/.031	**.730**/.214
TUD-Crossing	.695/.027	.374/.117	.640/.044	.260/.258	.695/.073	.059/.080	.712/.079	.459/.079	**.741**/.033	**.619**/.078
ETH-Bahnhof	.732/.020	.285/.078	.621/.033	.074/.256	.675/.046	-.101/.202	**.759**/.025	**.409**/.257	.757/.021	.292/.141
ETH-Jelmoli	.729/.029	.249/.131	.612/.030	-.005/.139	.664/.037	-.097/.212	**.757**/.035	.316/.132	**.757**/.029	**.350**/.199
ETH-Sunnyday	.754/.027	.386/.095	.643/.065	.126/.070	.686/.065	-.037/.485	.776/.039	.397/.075	**.777**/.030	**.603**/.190
Overall Average	.727/.032	.397/.170	.641/.047	.210/.271	.684/.055	.096/.373	.747/.049	.469/.211	**.751**/.036	**.474**/.236

TABLE VIII: Tracking results per detector on the public datasets averaged across the different trackers (DPF, RJMCMC, Hierarchy, SORT, and MDP). MOTA is tabulated as μ/σ, R/P stands for the detector Recall/Precision. **Best** and second best Recall, Precision, and MOTA are highlighted.

Dataset	**RCNN**		**DeepPed**		**LDCF**		**ACF**		**DPM**		**HOG-SVM**	
	R/P↑	**MOTA↑**	**R/P↑**	**MOTA↑**	**R/P↑**	**MOTA↑**	**R/P↑**	**MOTA↑**	**R/P↑**	**MOTA↑**	**R/P↑**	**MOTA↑**
CAVIAR-EnterExit	**.73**/.91	**.724**/.133	.60/.93	.542/.056	.71/**.97**	.323/.497	.69/.94	.476/.165	.69/.89	.588/.043	.62/.84	.480/.309
CAVIAR-OneShop	**.57**/.82	.326/.213	.21/.73	.092/.021	.45/.93	.139/.311	.43/**.95**	.160/.225	.51/.88	**.359**/.122	.37/.76	.147/.263
PETS-S2L1	.70/.79	.411/.285	.85/.88	.530/.295	**.92**/**.95**	**.673**/.102	**.92**/.94	.631/.139	.85/**.95**	.624/.136	.81/.89	.526/.212
TUD-Crossing	.70/.92	.386/.299	.69/.92	.440/.179	**.75**/**.93**	.381/.407	.74/**.93**	**.439**/.178	.63/.87	.391/.092	.53/.68	.087/.204
ETH-Bahnhof	.58/.75	.198/.288	.49/.57	.163/.091	**.70**/**.83**	**.305**/.273	.63/.78	.249/.193	.57/.73	.258/.113	.52/.47	-.022/.332
ETH-Jelmoli	.69/.81	.140/.313	.19/.79	.106/.150	.70/**.91**	**.298**/.218	.62/**.91**	.245/.136	**.76**/.85	.224/.112	.67/.60	-.037/.284
ETH-Sunnyday	.52/.76	.304/.332	.39/.71	.110/.072	.54/**.90**	.459/.382	.52/.87	.369/.225	**.56**/.79	**.498**/.166	.43/.52	.029/.415
Overall Average	.64/.82	.356/.321	.49/.79	.283/.245	**.68**/**.92**	.368/.364	.65/.90	.370/.238	.65/.85	**.421**/.185	.56/.68	.180/.352

TABLE IX: Tracking-by-detection performance comparison on mobile and static camera based datasets (averaged across each dataset category). **Best** and second best results in each category are shown; each result is reported as μ/σ.

Datasets		RCNN		DeepPed		LDCF		ACF		DPM		HOG-SVM		Average	
		MOTP↑	MOTA↑	MOTP↑	MOTA↑	MOTP↑	MOTA↑	MOTP↑	MOTA↑	MOTP↑	MOTA↑	MOTP↑	MOTA↑	MOTP↑	MOTA↑
Static Camera	**DPF**	.717/.033	.520/.162	.656/.044	.395/.198	.718/.036	.460/.165	.724/.035	.508/.160	.741/.016	.490/.095	.732/.017	.416/.132	.718/.039	.465/.159
	RJMCMC	.629/.028	.123/.218	.656/.019	.237/.214	.688/.028	.503/.092	.641/.049	.460/.108	.658/.022	.435/.052	.607/.057	.153/.229	.644/.047	.319/.224
	Hierarchy	.703/.027	.352/.327	.673/.010	.363/.269	.603/.061	-.188/.507	.691/.061	.252/.277	.731/.039	.393/.220	.705/.019	.186/.377	.687/.060	.226/.390
	SORT	.744/.041	.641/.093	.665/.006	.485/.222	.755/.020	.653/.155	.750/.055	.538/.205	**.766**/.017	.545/.120	.709/.091	.374/.289	.741/.057	**.539**/.211
	MDP	.743/.037	**.673**/.119	.669/.009	.524/.267	.759/.022	.467/.293	.757/.031	.376/.320	**.766**/.021	.590/.161	.747/.021	.483/.205	**.749**/.036	.519/.252
	Average	.707/.055	.462/.290	.666/.027	.401/.257	.704/.069	.379/.413	**.732**/.044	.427/.248	**.732**/.047	**.491**/.159	.700/.071	.322/.291		
Mobile Camera	**DPF**	.721/.003	.340/.071	.734/.010	.133/.036	.765/.016	.464/.097	.765/.013	.327/.066	.750/.014	.374/.123	.697/.019	.203/.104	.739/.028	.307/.139
	RJMCMC	.604/.006	-.289/.059	.646/.015	-.007/.040	.666/.032	.335/.075	.662/.032	.307/.057	.627/.019	.231/.076	.546/.018	-.186/.168	.625/.047	.065/.259
	Hierarchy	.686/.023	.032/.105	.720/.015	.071/.112	.590/.014	-.178/.057	.702/.014	.028/.094	.719/.024	.141/.119	.635/.006	-.562/.154	.675/.051	-.078/.262
	SORT	.748/.002	.494/.066	.768/.013	.176/.070	.787/.013	**.595**/.090	**.795**/.014	.310/.114	.785/.012	.446/.109	.701/.011	.222/.009	**.764**/.034	.374/.172
	MDP	.743/.001	.493/.090	.769/.012	.258/.032	.789/.015	.554/.203	.782/.017	.468/.247	.781/.009	.443/.211	.716/.014	.273/.110	**.764**/.029	**.415**/.197
	Average	.701/.056	.214/.323	.683/.172	.126/.114	.719/.084	**.354**/.313	**.741**/.056	.288/.200	.732/.062	.327/.184	.659/.067	-.010/.354		

the dataset, a detector or tracker change might help improve results.

Looking into which tracker, irrespective of the utilized detector, performs better (*Cf.* Q_2), the tracker choice evidently affects the performance of a given tracking-by-detection algorithm. This can easily be corroborated by looking at Table VII. Overall, based on average MOTA and MOTP, MDP stands out as the best tracker, irrespective of detector choice and context. It is followed by SORT and then DPF. This ranking is in line with the public results reported in the MOT-Challenge website [8]. In short, if one needs a good generic tracker, i.e., reliable whatever the context, MDP must be privileged. But if one wants a tracker that is not very sensitive to the choice of the detector, then it is necessary to privilege DPF (*Cf.* low σ of $.170$ across detector and dataset variations).

Like the tracker, the detector choice is very important (*i.e.,* Q_3). Even a small difference in detector recall rate can lead to a significant difference in tracking accuracy. For example, on TUD-Crossing, the best tracking accuracy is obtained with MDP-LDCF (76.3%) and second best with MDP-ACF (73%) – a 3.3% difference in accuracy even though there is only a 1% difference in recall between the two detectors (with the same precision). Based on Table VIII, we see that surprisingly DPM results in the best average (across datasets and trackers) tracking accuracy. The worst accuracies for all the trackers, mostly due to the high number of id switches, are obtained when using HOG-SVM. Even though LDCF exhibits the best average recall and precision on the datasets, DPM exhibits a 5.3% average tracking accuracy improvement over it. Hence, a better detector in detector benchmarking does not directly imply a better tracking accuracy. Depending on the number and dynamics of targets in the dataset, trackers can either improve detector performance (temporaly linking targets and thus filling in missed detections) or deteriorate it.

HOG-SVM on average has a tracking accuracy that is less than all the other detectors. Given that most state-of-the-art works report based on HOG-SVM like detectors [9], [1], it is possible to significantly improve those results by using any of the other detectors. The results also highlight that deep learning based detectors like RCNN and DeepPed, which are quite powerful and amongst the best detectors in the state-of-art in detection benchmarks [15], do not necessarily result in the best performance when coupled with trackers. Our results indicate better tracking accuracies are obtained when using DPM and LDCF detectors – a better detector does not guarantee the best performance when associated with a tracker. Additionally, DPM is the least sensitive to the choice of the tracker (see its standard deviation). LDCF is more sensitive to the choice of the associated tracker (see Table III) – it leads to the best tracking accuracy only when paired with the right trackers, i.e., SORT or MDP.

Looking at the performance of tracking-by-detection approaches on static and mobile datasets/contexts, i.e., Q_4, it is evident that the performance varies depending on the dataset/context (Table IX). Generally, a better average tracking accuracy is observed when dealing with fixed or static cameras than mobile ones. This is intuitive as the combined camera and target motion poses more challenge for trackers. In both static and mobile camera datasets, SORT and MDP, on average, stand out as the best trackers. DPM and LDCF detectors, on average, result in the best tracking accuracies on static and mobile camera datasets, respectively. Looking at individual tracking-by-detection approaches, on static datasets, MDP-RCNN and SORT-LDCF result in the best and second best tracking accuracies. On mobile datasets, this becomes SORT-LDCF and MDP-LDCF, respectively. As highlighted in the results section, the chosen detector type has more impact on tracking accuracy on mobile datasets than on static datasets. Even though the best performing trackers are MDP and SORT, DPF shows the least variability across the different detectors. It is more resilient to the detector choice. This is seconded by SORT.

The above discussions focus on the tracker accuracy as this is the metric most affected by detector choice. The tracker precision shows less variation. If we look at the trackers, when averaged over all datasets and detectors, on average there is an 11% difference between the best (MDP) and worst (RJMCMC) tracking precision results. The results also show small standard deviations, indicating less variability across dataset and detectors. If we look at the detectors, when averaged over all datasets and trackers, there is only a 9.7% and 8.1% average tracking precision loss between the best and worst detectors on static and mobile datasets respectively. On static datasets, the best precision is obtained when using DPM and worst when using DeepPed, on mobile datasets they are ACF and DeepPed for best and worst.

Based on the results and the above highlighted discussions, it is possible to outline the following important observations

and corresponding guidelines.

- MDP and SORT prove to be the best trackers (tracking performance and repeatability) on the seven datasets (*Cf.* Q_1 and Q_2). But, on the other hand, the ranking of the detectors is less clear and distributed over the different detectors: LDCF on three, DPM on two, and RCNN and ACF each on one datasets. *Hence, detector choice should be made more carefully specific to the application context.*
- The stochastic trackers (DPF and RJMCMC) inherently give less repeatable performance (*Cf.* Q_2). *Hence, if there is a strong constraint on repeatability, privilege non-stochastic trackers.*
- As demonstrated, and corroborated in the literature, e.g., [8], SORT shows very good performance on our evaluation datasets. This indicates a tracker with simple formulation can be sufficient for different application contexts. *Hence, more investigation efforts should be put on the choice and development of a detector adapted to the application context.*
- Our experimental results demonstrate that a 1% difference in detector recall could lead to as much as a 10% drop in MOTA (Table VIII and [58]). *Hence, the detector plays a key role in the global performance and more attention should be devoted to it.*
- Some of the detectors, i.e., LDCF and DPM, are more resilient to the tracker choice (*Cf.* Q_3). *Hence, if there is no control over the choice of a tracker, privilege these detectors.*
- LDCF and DPM detector coupled with either SORT or MDP are more robust to the diversity and variability of treated datasets. *Hence, privilege these detectors and trackers for a generic tracking-by-detection application, without any a-priori information about the nature of the scene.*
- Tracking-by-detection performance on mobile datasets is inferior compared to the ones on static datasets. This is due to the lower R/P detector performance (*Cf.* Q_4) and coupled camera and targets' motions. The detector choice has more impact on tracking accuracy on mobile datasets than on static datasets. *Hence, extra efforts should be taken to obtain better tracking results on mobile datasets, for example, by carefully selecting and tuning detectors for the application, and providing compensation for camera motion during tracking.*
- As discussed, tracking accuracy (MOTA) is a more relevant metric than tracking precision (MOTP). This is also iterated in the literature [8]. Additionally, our evaluations highlight the key role of detector in tracking-by-detection. *Since detectors are characterized by Recall/Precision (R/P), these metrics can also be used as tracking-by-detection performance indicators and more importantly should be used during tracking-by-detection system prototyping.*
- Our evaluation on these diverse set of datasets demonstrate Deep Learning (DL) based detectors under-perform when coupled with trackers. This is due to the inferior recall (missed targets) and precision (higher false positives), compared to LDCF and DPM, on most of the evaluation datasets – they are less generic. *Hence, even though DL detectors are powerful, the need for application specific training that entails (i) a large training dataset (tedious learning), and (ii) often dedicated hardware (GPU), makes them less appealing choices for tracking-by-detection.*

VIII. Conclusions

In this work, we have presented several tracking-by-detection comparative evaluations using a combination of five selected trackers and six detectors on seven public datasets. Our objective is not to develop a tracking-by-detection approach with the best absolute performance, but rather, to study (in term of relative performances) the influence of the detector and tracker choices have on overall tracking performance. The results show that the overall performance depends on how challenging the dataset is, the performance of the detector on the specific dataset, and the tracker-detector combination. Some trackers are more sensitive to the choice of detector and, reciprocating this, some detectors are also more sensitive to the choice of a tracker than others. The choice of the exact detector to use in tracking-by-detection should be carefully investigated, and if possible verified on a validation set before plugging into a tracker – state-of-the-art detector does not necessary lead to better tracking performance in all contexts. The need for careful evaluation needs to be further underscored given the recent advances in machine learning that are resulting in improved and very dynamic detection algorithms by the year.

Acknowledgment

This work was supported by grants from the French General Directorate for Armament (DGA) under grant reference SERVAT RAPID-142906073 and the Romeo2 project, (http://www.projetromeo.com/), BPIFrance in the framework of the Structuring Projects of Competitiveness Clusters (PSPC).

References

[1] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, L. Van Gool, Online multiperson tracking-by-detection from a single, uncalibrated camera, Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (9) (2011) 1820–1833.

[2] P. Dollár, R. Appel, S. Belongie, P. Perona, Fast feature pyramids for object detection, IEEE Trans. Pattern Anal. Mach. Intell. 36 (8) (2014) 1532–1545.

[3] P. Dollár, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: An evaluation of the state of the art, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (4) (2012) 743–761.

[4] X. Chen, Z. Qin, L. An, B. Bhanu, Multiperson tracking by online learned grouping model with nonlinear motion context, IEEE Transactions on Circuits and Systems for Video Technology 26 (12) (2016) 2226–2239.

[5] Z. Khan, T. Balch, T. Dellaert, Mcmc-based particle filtering for tracking a variable number of interacting targets, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (11) (2005) 1805–1918.

[6] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, M. Shah, Visual tracking: an experimental survey, Pattern Analysis and Machine Intelligence, IEEE Transactions on 36 (7) (2014) 1442–1468.

[7] E. Moussy, A. A. Mekonnen, G. Marion, F. Lerasle, A comparative view on exemplar "tracking-by-detection" approaches, in: IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS'15), 2015, pp. 1–6.

[8] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, K. Schindler, MOTChallenge 2015: Towards a benchmark for multi-target tracking, arXiv:1504.01942 [cs]ArXiv: 1504.01942.
URL http://arxiv.org/abs/1504.01942

[9] J. Zhang, L. L. Presti, S. Sclaroff, Online multi-person tracking by tracker hierarchy, in: IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS'12), Beijing, China, 2012.

[10] A. Bewley, Z. Ge, L. Ott, F. Ramos, B. Upcroft, Simple online and realtime tracking, in: IEEE International Conference on Image Processing (ICIP'16), 2016, pp. 3464–3468.

[11] Y. Xiang, A. Alahi, S. Savarese, Learning to track: Online multi-object tracking by decision making, in: IEEE International Conference on Computer Vision (ICCV'15), 2015.

[12] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005.

[13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (9) (2010) 1627–1645.

[14] W. Nam, P. Dollár, J. H. Han, Local decorrelation for improved pedestrian detection, in: Advances in Neural Information Processing Systems (NIPS'14), 2014, pp. 424–432.

[15] R. Girshick, J. Donahue, T. Darrell, J. Malik, Region-based convolutional networks for accurate object detection and segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (1) (2016) 142–158.

[16] D. Tom, F. Monti, L. Baroffio, L. Bondi, M. Tagliasacchi, S. Tubaro, Deep convolutional neural networks for pedestrian detection, arXiv preprint arXiv:1510.03608.

[17] J. Berclaz, F. Fleuret, P. Fua, Robust people tracking with global trajectory optimization, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06), Vol. 1, 2006, pp. 744–750.

[18] D. Gernimo, F. Lerasle, A. Lpez, State-driven particle filter for multi-person tracking, in: Advanced Concepts for Intelligent Vision Systems (ACIVS'12), Brno, Czech Republic, 2012, pp. 467–478.

[19] S.-H. Bae, K.-J. Yoon, Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'14), 2014, pp. 1218–1225.

[20] D. Beymer, K. Konolige, Real-time tracking of multiple people using continuous detection, in: IEEE International Conference on Computer Vision (ICCV'99), 1999.

[21] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, D. G. Lowe, A boosted particle filter: Multitarget detection and tracking, in: Computer Vision-ECCV 2004, Springer, 2004, pp. 28–39.

[22] M. Andriluka, S. Roth, B. Schiele, People-tracking-by-detection and people-detection-by-tracking, in: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE, 2008, pp. 1–8.

[23] D. Gerónimo, A. López, A. Sappa, T. Graf, Survey of pedestrian detection for advanced driver assistance systems, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (7) (2010) 1239–1258.

[24] P. A. Viola, M. J. Jones, Robust real-time face detection, International Journal of Computer Vision 57 (2) (2004) 137–154.

[25] X. Wang, T. Han, S. Yan, An HOG-LBP human detector with partial occlusion handling, in: IEEE International Conference on Computer Vision (ICCV'09), Kyoto, Japan, 2009.

[26] C. Wojek, B. Schiele, A performance evaluation of single and multi-feature people detection, in: DAGM-Symposium, Munich, Germany, 2008, pp. 82–91.

[27] P. Dollár, Z. Tu, P. Perona, S. Belongie, Integral channel features, in: British Machine Vision Conference (BMVC'09), London,UK, 2009.

[28] C. Zhang, P. A. Viola, Multiple-instance pruning for learning efficient cascade detectors, in: Advances in Neural Information Processing Systems (NIPS'08), 2008, pp. 1681–1688.

[29] L. Bourdev, J. Brandt, Robust object detection via soft cascade, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR'05), Washington, DC, USA, 2005, pp. 236–243.

[30] S. Zhang, R. Benenson, B. Schiele, Filtered channel features for pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15), 2015.

[31] K. Smith, D. Gatica-Perez, J. M. Odobez, Using particles to track varying numbers of interacting people, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005.

[32] M. Isard, J. MacCormick, Bramble: a bayesian multiple-blob tracker, in: International Conference on Computer Vision (ICCV'01), Vol. 2, Vancouver, Canada, 2001.

[33] P. Perez, J. Vermaak, A. Blake, Data dusion for visual tracking with particles, Proceedings of the IEEE 92 (3) (2004) 495–513.

[34] M. Luber, J. Stork, G. Tipaldi, K. Arras, People tracking with human motion predictions from social forces, in: International Conference on Robotics and Automation (ICRA'10), Anchorage, AK, USA, 2010.

[35] D. A. Ross, J. Lim, R.-S. Lin, M.-H. Yang, Incremental learning for robust visual tracking, International Journal of Computer Vision 77 (1-3) (2008) 125–141.

[36] X. Mei, H. Ling, Robust visual tracking and vehicle classification via sparse representation, Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (11) (2011) 2259–2272.

[37] S. Avidan, Support vector tracking, Pattern Analysis and Machine Intelligence, IEEE Transactions on 26 (8) (2004) 1064–1072.

[38] B. Babenko, M.-H. Yang, S. Belongie, Visual tracking with online multiple instance learning, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 983–990.

[39] M. Isard, A. Blake, Icondensation: Unifying low-level and high-level tracking in a stochastic framework, in: Computer VisionECCV'98, Springer, 1998, pp. 893–908.

[40] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, L. Van Gool, Robust tracking-by-detection using a detector confidence particle filter, in: International Conference on Computer Vision (ICCV'09), Kyoto, Japan, 2009.

[41] S. Zhang, R. Benenson, M. Omran, J. H. Hosang, B. Schiele, How far are we from solving pedestrian detection?, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16), Las Vegas, NV, USA, 2016, pp. 1259–1267.

[42] R. Benenson, M. Omran, J. Hosang, B. Schiele, Ten years of pedestrian detection, what have we learned?, in: Computer Vision - ECCV 2014 Workshops, Vol. 8926, 2015, pp. 613–627.

[43] J. yves Bouguet, Pyramidal implementation of the lucas kanade feature tracker, Intel Corporation, Microprocessor Research Labs.

[44] K. C. Smith, Bayesian methods for visual multi-object tracking with applications to human activity recognition, Ph.D. thesis, Ecole Polytechnique Federale de Lausanne (Switzerland) (2007).

[45] W. K. Hastings, Monte carlo sampling methods using markov chains and their applications, Biometrika 57 (1) (1970) 97–109.

[46] H. W. Kuhn, The hungarian method for the assignment problem, Naval Research Logistics Quarterly 2 (1-2) (1955) 83–97.

[47] K. Bernardin, R. Stiefelhagen, Evaluating multiple object tracking performance: the CLEAR MOT metrics, EURASIP Journal on Image and Video Processing 2008 (2008) 1:1–1:10.

[48] A. A. Mekonnen, F. Lerasle, Comparative evaluations of selected tracking-by-detection approaches: Supplemental material, homepages.laas.fr/lerasle/pdf/supp_material.pdf (2017).

[49] CAVIAR-Project, CAVIAR test case scenarios (2004).
URL http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/

[50] J. Ferryman, A. Shahrokni, Pets2009: Dataset and challenge, in: IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, 2009, pp. 1–6.

[51] M. Andriluka, S. Roth, B. Schiele, Monocular 3d pose estimation and tracking by detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10), 2010.

[52] A. Ess, B. Leibe, K. Schindler, , L. van Gool, Robust multiperson tracking from a mobile platform, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (10) (2009) 1831–1846.

[53] P. Dollár, Piotr's Image and Video Matlab Toolbox (PMT), http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html.

[54] R. B. Girshick, P. F. Felzenszwalb, D. McAllester, Discriminatively trained deformable part models, release 5, http://people.cs.uchicago.edu/~rbg/latent-release5/.

[55] OpenCv, Opencv: Open source computer vision library, http://www.opencv.org.

[56] P. Pérez, C. Hue, J. Vermaak, M. Gangnet, Color-based probabilistic tracking, in: European Conference on Computer Vision (ECCV'02), Copenhagen, Denmark, 2002.

[57] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The pascal visual object classes (VOC) challenge, International Journal of Computer Vision 88 (2) (2010) 303–338.

[58] A. A. Mekonnen, F. Lerasle, Comparative evaluations of selected tracking-by-detection approaches: Raw experimental evaluation results, homepages.laas.fr/lerasle/pdf/raw_results.xls (2017).