Revisiting Jump-Diffusion Process for Visual Tracking: A Reinforcement Learning Approach

Xiaobai Liu*, Qian Xu*, Thuan Chau, Yadong Mu, Lei Zhu, Shuicheng Yan, IEEE Fellow

Abstract—In this work, we revisit the classical stochastic jump-diffusion process and develop an effective variant for estimating visibility statuses of objects while tracking them in videos. Dealing with partial or full occlusions is a long standing problem in computer vision but largely remains unsolved. In this work, we cast the above problem as a Markov Decision Process and develop a policy-based jump-diffusion method to jointly track object locations in videos and estimate their visibility statuses. Our method employs a set of jump dynamics to change object's visibility statuses and a set of diffusion dynamics to track objects in videos. Different from traditional jump-diffusion process that stochastically generates dynamics, we utilize deep policy functions to determine the best dynamic for the present state and learn the optimal policies using reinforcement learning methods. Our method is capable of tracking objects with full or partial occlusions in crowded scenes. We evaluate the proposed method over challenging video sequences and compare it to alternative tracking methods. Significant improvements are made particularly for videos with frequent interactions or occlusions.

1 Introduction

Tracking objects of interest [51] in videos has attracted many interests in the past decade since it has broad potentials in multiple applications, e.g., security surveillance, autonomous driving, border control, disaster response, and forensic video analysis, etc. Object trajectories, once estimated, can be used to generate high-level video understandings, e.g., activities, events. However, a long standing research problem is how to track objects that are partially visible or completely invisible due to occlusions. Classic solutions [38], [35], [25] to this challenge often employ a hypothesis-testing strategy to determine object visibility status: assuming that the object of interest is visible, matching image of object (or object part) into the next video frame, and checking if the matching score is higher than a threshold. This strategy has two fundamental shortcomings. First, the optimal threshold might be varying across time, locations, objects, scene types, and it is very difficult to find the optimal one while tracking an object moving in the scene. Second, these methods ignore the time-dependent relationships between the visibility statuses of the same object. As shown in Figure 1, as the human A approaches and enters the vehicle B, the visibility status of B changes over time, form visible, occluded, to visible. To our best knowledge, there is no previous efforts on systematically studying the temporal transitions of object visibility status. In this work, we fill in the gap through developing an unified method that can explicitly reason object visibility statuses while tracking objects in videos.

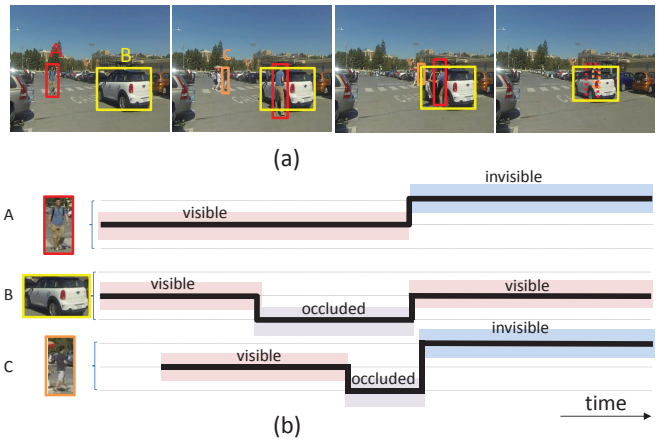

Fig. 1. Object tracking with occlusions. (a) Three objects (pedestrians or vehicles) in the same scene are interacting with each other. (b) The visibility status of each object is switching between visible, occluded, and invisible over time.

The key idea of our method is to formulate object tracking as a Markov Decision Process in a joint continuous-discrete space. For each object, our method aims to estimate its visibility status as well as its location and size in each video frame. The estimation of visibility status is defined in a discrete space composed of 'visible', 'occluded' and 'invisible ', and the task of localization is defined in a continuous image space, i.e. location coordinates. On the one hand, these two tasks can be independently formulated and separately solved. For example, we can learn a discriminative model to predict the visibility status of a vehicle given its images, or learn to decide the optimal bounding box for this object. On the other hand, the two tasks should be coordinately formulated and scheduled in order to avoid errors in visual tracking. When the object of interest becomes invisible in videos, for example, traditional trackers do not work and we need to employ re-identification methods [55], [49] to group discrete trajectories

*The first two authors contributed equally to this work. **Xiaobai Liu** is with the Department of Computer Science, San Diego State University, San Diego, California 92125*
***Qian Xu** is with the XreLab Inc., San Diego, California 92128*
***Thuan Chau** is with the Department of Computer Science, San Diego State University, San Diego, California 92125*
***Lei Zhu** is with the School of Information Science and Engineering, Shandong Normal University, Jinan, China 250358*
***Yadong Mu** is with the Institute of Computer Science and Technology, Peking University, Beijing, China 100000*
***Shuicheng Yan** is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore*

together. Therefore, a robust tracker is expected to jointly estimate object visibility statuses and object locations in video frames.

As the main innovation of this effort, we develop a policy-based jump-diffusion process [20] to seek for the optimal solution in the continuous-discrete space. The process starts with an initial solution, i.e., visibility status and locations, and iteratively employs a set of dynamics to reconfigure the present solution in order to simulate a Markov Chain in the solution space. A dynamic is either a jump that changes object visibility statuses or a diffusion that changes object location/size in videos. In the literature [20], [56], a traditional jump-diffusion process is often driven by randomly proposed dynamics or data-driven dynamics, and is subject to slow convergence. In this work, we focus on the real-time visual tracking problem, for which convergence speed is the most critical factor. To address this issue, we propose to employ discriminatively trained policy functions to propose dynamics and parameterize the policy functions using deep neural networks [6], [32]. We train policy networks using the policy gradient method [32] in the reinforcement learning setting. Evaluations on multiple video benchmarks showed that our method can reliably estimate object visibility status over time and robustly track objects of interest with occlusions in crowded scenarios.

The contributions of this work include (I) An unified tracking method that can explicitly reason object visibility statuses in crowded scenes; and (II) a policy-based jump-diffusion method that employs discriminatively trained policy functions to drive the simulation process. These techniques can be applied to address the challenges of other compute vision tasks.

2 Relationships to Previous Works

The proposed research is closely related to the following research streams in computer vision and AI.

Object Tracking In past literature, tracking-by-detection has become the mainstream framework [45], [9]. Specifically, a general detector [12], [36] is first applied over video frames to obtain detection proposals, then data association techniques [2], [8], [54] are employed to link detection proposals over time in order to get object trajectories. The advances in object detection, e.g., [26] [27], significantly improve detectors' generalization capabilities and robustness while dealing with small objects, scene noises, and other challenges. Our approach follows the same tracking-by-detection strategy but focuses on the reasoning of object visibility status.

Occlusion Handling is perhaps the most fundamental problem in visual tracking, and has been extensively studied in the past literatures. These methods can be roughly divided into three categories: i) using depth information [11], [52], ii) modeling partial occlusion relations [41], iii) modeling object appearing and disappearing globally [43], [44], [29]. As aforementioned, these methods employ the hypothesis-test strategy to determine if occlusions happen, which is not reliable. More recently, Qi et al. [7] introduced a visual attention map to characterize target visibility (occlusion) statuses, and used it to control the online model updating process. This method quantizes occlusion statuses in a continuous space and aims to regress these real-valued confidences from low-level features. In contrast, this paper represents occlusion statuses in a discrete space and presents a principled way to explicitly reason object visibility status while tracking them in videos.

Reinforcement Tracking Reinforcement learning methods [40] aim to learn to sequentially choose actions that can maximize cumulative future rewards. Traditional reinforcement learning methods are limited to their poor performance and stability capabilities. The recent deep reinforcement learning (DRL) methods integrated deep representations to parameterize policy functions or value functions, and achieved encouraging successes in multiple fields, e.g., video games [31], or Go [32]. Recently, DRL was also applied to solve traditional computer vision problems. In particular, Caicedo and Lazebnik [4] employed DRL to narrow down search areas while localizing objects, Yun et al. [53] extended this idea to track objects in videos. Choi et al. [6] employed policy functions to determine the best template while tracking multiple objects in videos. Huang et al. [21] employed policy functions to classify video frames to be either easy or difficult cases and adaptively use shallow or deep models to deal with them in order to boost system efficiencies. James and Deva [39] developed an interactive video processing system which utilizes policy functions to collect human annotations with minimal interactions and to estimate the status of tracks (i.e. being lost or not). These methods achieved impressive results on public video benchmarks, and demonstrated the effectiveness of DLR methods. In this work, we extend these works and introduce an tracking framework to explicitly reason object occlusion statuses. We also contribute an effective variant of jump-diffusion process that has wide application potentials in multiple computer vision fields.

Our approach will collect both positive and negative samples to train the proposed policy functions. Every sample contribute equally to the training procedure. This strategy could be improved by discriminatively assigning varying weights to each training sample, as suggested by Ma et al. [28]. Our tracking method can be integrated with advanced video analysis techniques, e.g., semantic pooling by Chang et al. [5], to further accelerate inference time. In this work, we focus on the study of learning-driven sampling algorithm and the applications of the above techniques are out of the scope.

3 The Proposed Method

3.1 Background: stochastic jump-diffusion process

Stochastic jump-diffusion process was first studied by Grenander and Miller [18] in 1994 and was rigorously formulated by Green [17] in 1995. The process is often used to sample a probabilistic distribution defined over a mixture space composed of multiple subspaces of varying dimensions. It employs both jump and diffusion dynamics to reconfigure the present sample, and simulates a Markov Chain towards the target distribution. A jump is used to move between different subspaces whereas a diffusion is used to move within the same subspace. While this method was barely used in computer vision community, the seminal works by Song-Chun Zhu et al. [56] [20] showed that it can serve as a unified inference framework for solving most vision tasks, e.g., image segmentation, skeleton detection etc. The traditional jump-diffusion process, however, uses data-driven proposals or random proposals to change the present sample and is not applicable for real-time visual systems, e.g., object tracking. In this work, we revisit jump-diffusion process and study an effective proposal making strategy that enables the applications of this process over visual tracking.

Algorithm 1: Policy-based Jump-Diffusion Tracker (JD-Tracker).

Input: video sequences
Initializations: object detections
while *Not Convergence* **do**
- Call the discrete policy network to generate the conditional action probabilities;
- Execute a stochastically selected discrete action;
- If the current status is invisible, terminate;
- Call the continuous policy network to generate the conditional action probabilities;
- Execute a stochastically selected continuous action;

end

3.2 Overview of the proposed method

The objective of this work is to track objects of interest in video sequences and simultaneously estimate their visibility statuses over time. We formulate this joint task as a policy-based jump-diffusion process, which employs a set of dynamics to simulate a Markov Chain traveling towards the optimal solution. As the object of interest moves in videos, our method employs two type of dynamics to reconfigure the current solution: jump dynamics, that change the discrete object visibility status, and diffusion dynamics, that change the continuous object location/size. We employ policy functions to seek for the optimal dynamic in either discrete or continuous space, and parameterize the policy functions using deep neural networks. A reinforcement method is developed to train these networks from annotated video data. The learned policy functions are used to make proposals while simulating the Markov Chain. We will empirically show that the proposed method can robustly track objects in crowded scenes with frequent occlusions, and accurately estimate object visibility statuses.

3.3 Policy-based jump-diffusion tracking

Algorithm 1 summarizes the sketch of the proposed policy-based jump-diffusion tracker. For a given video sequence, we first detect objects of interest, and for each object detection, randomly chooses a jump or diffusion dynamic to change its visibility status or locations. Note that if the object of interest is invisible, the algorithm terminates and the tracking of this object ends.

A critical problem for this jump-diffusion tracker is how to select an appropriate dynamic at each step. In this work, we consider the input video sequences as the environment, and the developed tracker as an agent. In each video frame, the agent employs policy functions and current state information to retrieve the optimal action, execute the action to get a new state, and receive a reward. The agent seeks for optimal actions to change object visibility status or localize/resize the object while maximizing the accumulated future rewards.

We cast the selection of dynamics a Markov Chain Decision Process (MDP). An MDP is defined by a tuple $(s, a, \mathcal{T}, r)$, where $s \in S$ represents an object state, $a \in A$ denotes an action, $\mathcal{T}$ represents the transition between states, and $r(s, a)$ defines the immediate reward received after executing action a over a state s.

State variables are used to characterize the current status of an object in the present environment. For each object, the state space comprises of two subspaces, $S = S_d \cup S_c$, where S_d includes three discrete states, i.e., visible, occluded, and invisible, and S_c represents the continuous object positions and sizes in video frames. Every state represents the information of the target object, in terms of appearances, locations, and history of states.

Actions and Transition Function can be performed over objects to change their discrete or continuous states. In the discrete space, an action represents a jump dynamic and is applied to set the object's visibility status to be visible, occluded, or invisible. In the continuous space, an action represents a diffusion dynamic and is applied to shift or resize the bounding box of the object to cover it in the new video frame. A continuous action can be denoted with a triplet (θ, β, σ), representing the translation and scaling changes over orientation, distance, and size. Figure 2 visualizes this vector in a log-polar coordination. This action space is different from the previous works [53], [4], which quantize the translation and scaling spaces into discrete bins and introduce a classification network to define the policy function. In contrast, we define continuous actions in a 3-dimensional space and introduce a regression network to parameterize the policy function.

Reward function $r(s, a)$ is defined over the present state s, regardless of the actions a, i.e., $r(s, a) = r(s)$. In the discrete space, we apply an action to change the visibility status of the object of interest, and the reward function $r(s)$ returns 1 if the new visibility status is equal to the ground-truth status; or returns 0 otherwise. In the continuous space, we use an action to translate or re-size the target bounding box, and calculate the overlapping ratio between the new bounding box and ground-truth box, denoted as U. We set the reward function $r(s)$ as:

$$r(s) = \begin{cases} +1, U > 0.7 \\ -1, U < 0.5 \qquad \forall s \in S_c \\ 0, \quad Otherwise \end{cases} \tag{1}$$

where the two constants 0.7 and 0.5 are empirically set. We use these two thresholds throughout the experiments of this work.

Policy Functions The policy function $\pi(a_t|s_t)$ in MDP is used to choose a proper action a_t given the state s_t at time t. The action can be chosen in either deterministic or stochastic manner. While it is possible for a policy function to take any form, recent studies [32] [53] showed it is beneficial to parameterize it using deep neural networks, which can significantly improve the stability in policy learning. In this work, we employ deep networks to define policy functions in both continuous and discrete spaces, as introduced in Section 3.4.

Heuristic Policies As an alternative to policy functions, the simplest approach to selecting actions might be to pre-define heuristics. In the discrete action space, for example, one can set an object to be visible if it can be detected with high confidence, or to be occluded if only object parts are detectable (with high confidences), or to be invisible if neither objects or object parts are detectable. One can also evaluate the history of object movements, and set the visibility status of an object to be occluded if another object will walk in front of it. Similarly, in the continuous space, one can employ the traditional trackers to determine the translation or scale changes of object bounding box. The above heuristics can be further enhanced through using the ground-truth data, instead of detectors or trackers. In this work, we use these heuristics to initialize the learning of policy networks.

3.4 Architecture of Deep Policy Networks

We employ deep neural networks to parameterize the policy functions and use them to choose the optimal action for the

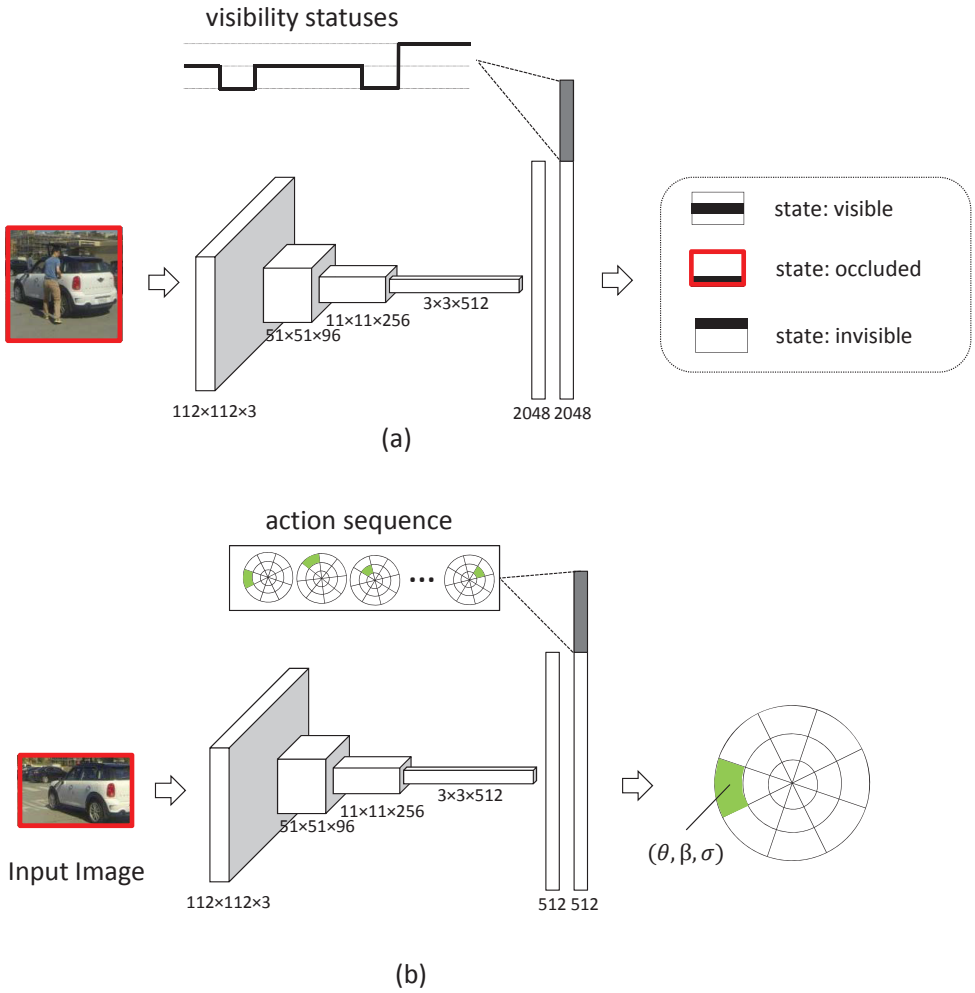

Fig. 2. Architectures of policy networks. (a) Policy network that maps an input image to one of the three visibility statuses: visible, invisible, and occluded. (b) Policy network that maps an input image to a continuous action used to shift or re-size object bounding boxes. Each continuous action is described as a triplet (θ, β, σ) that represents the translation direction, translation distance and scale factors, respectively. The two networks share the same convolutional layers.

current state. Figure 2 illustrates the network architectures used for continuous and discrete policy functions.

Discrete Policy Network Figure 2 (a) summarizes the sketch of the policy network used for selecting discrete actions. To determine the visibility status of a tracked object at time t, we use the bounding box at time $t-1$ to crop the image of object from the video frame at t, and feed it to a deep neural network. The network comprises of three convolution layers and two fully connected (fc) layers that are combined with the ReLu and dropout layers. The output of the second fc layer is concatenated with the vector of past visibility statuses which has 100 dimensions. The final layers are used to predict the probability of each visibility status of the given state. In particular, there are three output units, representing the 3 visibility statuses: visible, occluded and invisible, being combined with a softmax layer whose output is the conditional action probability distribution.

Continuous Policy Network We employ a similar network architecture to parameterize the policy network in continuous spaces, as shown in Figure 2 (b). The network shares the identical convolutional layers with the discrete policy network, and includes two additional fully connected layers that are combined with the ReLu and dropout layers. The output of the second fc layer is concatenated with the vector of past actions which has 100 dimensions. The final layer includes a single unit connected with a regression loss. We use the sum of square loss in this work. Note that this is different from the previous work [53] which employs classification loss to predict translation/scaling changes and ignores the continuous relationships between actions.

3.5 Training of Policy Networks

We formulate the learning of policy networks in the reinforcement setting. During training, the agent (tracker) will receive a reward signal from the environment (input video sequences in this work) after executing an action at time t. The training objective is to maximize the total rewards the agent can receive in the future. Note that the two policy networks are trained alternatively so that the developed jump-diffusion tracker can choose either continuous or discrete actions for the given state.

We use a variant of REINFORCE algorithm [46] with accumulated policy gradients to learn the parameters of the two policy networks. To do so, we randomly generate multiple tracking episodes with varying lengths (frames) from training videos, such that each episode covers the lifespan of an object of interest.

Algorithm 2: Training of Policy Networks.

Input: an episode of video frames and annotations
for *every video frame at* t **do**
-Use the policy networks to generate conditional probabilities for discrete actions and continuous actions, respectively;
-Choose and execute an action with a probability;
-Accumulate gradients ΔW according to Eq. (2);
end
if *tracking the object successfully* **then**
-Update network parameters $W = W + \Delta W$
else
-Update network parameters $W = W - \Delta W$;
end

Then, we perform object tracking on each training episode and draw actions as follows: employing the current policy reworks to generate the conditional action probabilities in both discrete and continuous spaces; stochastically selecting one of the actions as the current decision. In this way, we can probabilistically generate action sequences from the episode. Once selected the action, being discrete or continuous, we consider it as the ground-truth label and run the back-propagation method to calculate gradients. Taking the discrete policy network for an example, let $\phi(\alpha_t|s_t; W)$ denote the policy function and W the network parameters. We can accumulate the gradients for all decisions in the same episode as:

$$\Delta W = c \sum_{t=1}^{T} \Delta \log \pi(\alpha_t|s_t; W)\gamma^{T-t} \tag{2}$$

where c is the learning rate, Δ represents the derivative with respect to the network parameters W, T is the length of the episode and $\gamma \in (0, 1]$ is a discounting factor that assigns higher weight to decisions made later during the course of the episode. Similar gradient calculations are applied for the continuous policy network. We use ΔW to update network parameters when our tracker successfully localizes the object by the end of the episode. Otherwise, if our tracker fails to localize the object, we use negative gradients, i.e. $-\Delta W$, to update the network parameters. Algorithm 2 summarizes the sketch of the proposed training algorithm.

4 Experiments

4.1 Datasets, Implementation, and Metrics

We apply the proposed method over three video video datasets for test and evaluation purposes and compare it to the alternatives.

Tracking Interacting Objects (TIO) dataset. We collect a new video dataset to justify the effectiveness of the proposed occlusion reasoning method. The videos are captured in a parking-lot, and include multiple human-object interactions, e.g., loading, unloading, stopping etc. The objects of interest include pedestrians and vehicles. In contrast, most existing tracking benchmarks, e.g., PETS09 [13], OTB [47], KITTI dataset [16], do not include frequent occlusions. All video sequences are captured by a GoPro camera, with frame rate 30fps and resolution 1920×1080. The total number of frames of TIO dataset is more than 30K. There exist severe occlusions and large scale changes, making this dataset very challenging for traditional tracking methods.

People-Car dataset [43][1]. This dataset consists of video sequences on a parking lot with two synchronized bird-view cameras, with length $300 - 5100$ frames. In this dataset, there are many instances of people getting in and out of cars. This dataset is challenging because of frequent subject-object or subject-subject interactions, light variations and low object resolutions. It includes 22 video sequences in a plaza where people walk around and get in/out vehicles, and 3 video sequences in a parking lot that show vehicles entering/exiting the parking lot, people getting in/out vehicles, people interacting with trunk/suitcase.

PPL-DA dataset. We collect another video dataset to cover people's daily activities. This dataset consists of 3 public facilities: foot court, office reception, and plaza. The scenes are recorded with 4 GoPro cameras, mounted on around 1.5 meters high tripods. The produced videos are also around 4 minutes long and in 1080P high quality.

Beside the above testing data, we collect another set of video clips for training. To avoid over-fitting, we set up different camera positions, different people and vehicles from the testing settings. The training data consists of 380 video clips covering 9 events: *walking, opening vehicle door, entering vehicle, exiting vehicle, closing vehicle door, opening vehicle trunk, loading baggage, unloading baggage, closing vehicle trunk*. Both the datasets and short clips are annotated with the bounding boxes for people, suitcase, and vehicles. We additionally annotate the visibility status, i.e. "visible", "occluded", and "invisible", for the TIO dataset. We utilize the video annotation toolkit VATIC [42] to annotate the videos.

Implementation We pre-train the policy networks on the ImageNet dataset [10], and then fine-tune network parameters on the randomly generated episodes. We resize all images to be 112×112 pixels before feeding into the policy networks. We use dropout regularization for fully-connected layers, with drop rate 0.7. Each convolution layer is followed by the rectified linear unit (ReLU) activation function. To train the network, we set the learning rate c to be 0.0001, $\gamma = 0.95$. We randomly generate 6000 episodes from the training dataset, with length varying from 30 to 300 frames. We consider a predicted object box to be correct if its overlapping ratio with the ground-truth box is at least 0.5, and consider an episode to be successfully tracked if at last 40% predicted bounding box are correct. At the the early stage of training, we use the heuristic policies, instead of the policy networks, to choose actions, in order to accelerate the exploration stage. We use experience replay method [37] during training and retain in the replay memory 5000 successful samples and 5000 failure samples. To update the network parameters, we sample 50 samples from the memory and accumulate the gradients. It is noteworthy that we used the same policy representation for both vehicles and pedestrians and use the same two policy networks for tracking different categories. This is feasible because of the high generalization capabilities of deep neural networks.

We implement the proposed tracker using MatConvNet toolbox and run all experiments on a workstation with CPU: Intel Core i7-7700K, GPU: Nvidia GeForce GTX 1050, and Memory: 8GB. Without code optimization, the proposed tracker can run about 30 fps.

Metrics We evaluate the proposed method from two aspects. *First*, the ability to track objects. We adopt the widely used CLEAR metrics [22] to measure the performances of tracking

1. This dataset is available at cvlab.epfl.ch/research/surv/interacting-objects

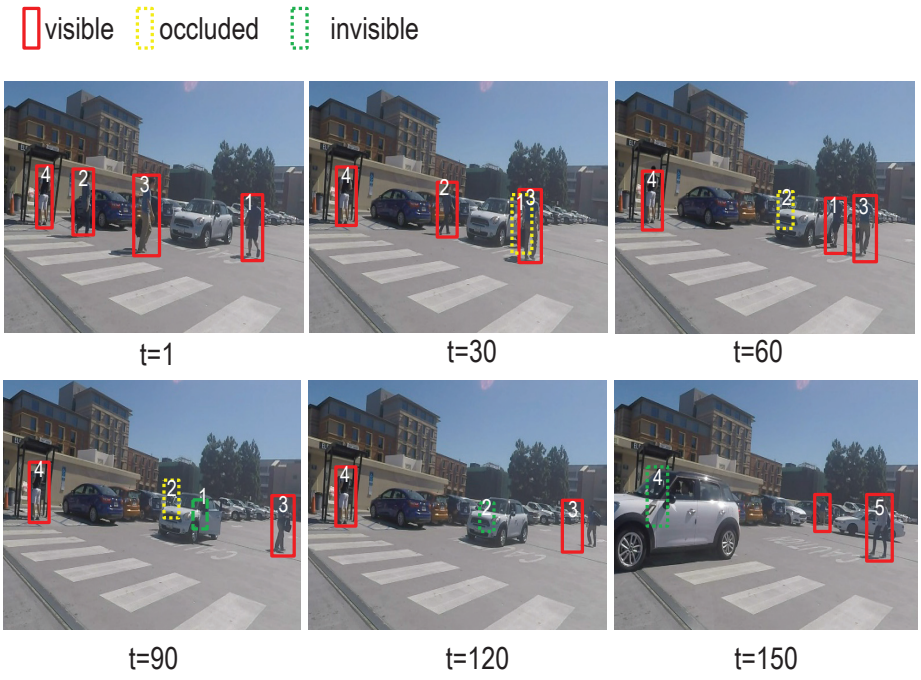

Fig. 3. Quantitative results of visibility status estimation on **TIO dataset**. t: index of video frames.

Scene	JDTracker	DCNet
Plaza	0.81	0.47
Parking-lot	0.73	0.29
Average	0.77	0.38

TABLE 1
Results of visibility status recognition (accuracy) on the **TIO** dataset. JDTracker: the proposed method; DCNet: the baseline method based on deep networks.

methods. It includes four metrics, i.e., Multiple Object Detection Accuracy (MODA), Detection Precision (MODP), Multiple Object Tracking Accuracy (MOTA) and Tracking Precision (MOTP), which take into account three kinds of tracking errors: false positives, false negatives and identity switches. We also report the number of false positives (FP), false negatives (FN), identity switches (IDS) and fragments (Frag). A higher value means better performance for TA and TP while a lower value means better performance for FP, FN, IDS and Frag. If the Intersection-over-Union (IoU) ratio of tracking results to ground truth is above 0.5, we accept the tracking result as a correct hit. The other metrics used include **MT**, *mostly tracked*, percentage of ground truth trajectories which are covered by tracker output for more than 80% in length; and **ML**$^{\downarrow}$, *mostly lost*, percentage of ground-truth trajectories which are covered by tracker output for less than 20% in length. *Second*, the ability to estimate object visibility status. We simply compare the prediction to the ground-truth labels and calculate the percentage of correctness, i.e. accuracy.

4.2 Results for Visibility Status Estimation

We apply the proposed method, denoted as **JDTracker**, over TIO dataset and test its ability to estimate object visibility status. Figure 3 visualizes the results on a video sequence. There are four pedestrians walking in a parking-lot and a vehicle moving in the nearby area. Taking the pedestrian$\#2$ for an instance, his visibility status changes from 'visible' (t=1, t=30) to 'occluded' (t=60, t=90) and 'invisible' (t=120) as he approaches and gets into the vehicle. Our method can correctly estimate the changes of visibility and enable high-level video understanding, e.g., activity recognition.

Table 1 reports the quantitative results of visibility status estimation while applying the proposed tracker over the TIO videos. For comparisons, we include the results of a baseline method, DCNet, which takes as input the image of object and outputs its discrete visibility labels. The network architecture of DCNet is the same as the discrete policy network but does not include the history of visibility statuses as inputs. We train DCNet in the fully supervised setting using the same training data. As shown in the table, the proposed tracker can achieve much better accuracies than the baseline. These comparisons are used to demonstrate the basic observation, as aforementioned, that visibility statuses of the same object are strongly time-dependent and it is important to make prediction using the previous visibility statuses.

4.3 Results for Object Tracking

For the **TIO** dataset, we compare the proposed method (denoted as **JDTracker**) with 6 state-of-the-arts: successive shortest path algorithm (SSP) [34], multiple hypothesis tracking with distinctive appearance model (MHT_D) [23], Markov Decision Processes with Reinforcement Learning (MDP) [48], Discrete-Continuous Energy Minimization (DCEM) [30], Discrete-continuous optimization (DCO) [1] and Joint Probabilistic Data Association (JPDA_m) [19]. We use the public implementations of these methods. Table 2 reports the quantitative results and comparisons on TIO dataset. From the results, we can observe that our method achieved encouraging performance while using most metrics. In particular, the IDS of our method is much lower than any other methods. This indicates that the ability of occlusion reasoning can

Plaza	MOTA ↑	MOTP ↑	FP ↓	FN ↓	IDS ↓	Frag ↓
JDTracker	**51.0%**	**78.3%**	5	**349**	**1**	3
MHT_D [23]	34.3%	73.8%	56	661	15	18
MDP [48]	32.9%	73.2%	24	656	9	7
DCEM [30]	32.3%	76.5%	**2**	675	2	**2**
SSP [34]	31.7%	72.1%	19	678	21	25
DCO [1]	29.5%	76.4%	22	673	6	2
JPDA_m [19]	13.5%	72.2%	163	673	6	3
ParkingLot	MOTA ↑	MOTP ↑	FP ↓	FN ↓	IDS ↓	Frag ↓
JDTracker	**39.3%**	**79.8%**	**321**	**1531**	**8**	**6**
MDP [48]	30.1%	76.4%	397	2296	26	22
DCEM [30]	29.4%	77.5%	383	2346	16	15
SSP [34]	28.9%	75.0%	416	2337	12	14
MHT_D [23]	25.6%	75.7%	720	2170	15	12
DCO [1]	24.3%	78.1%	536	2367	38	10
JPDA_m [19]	12.3%	74.2%	1173	2263	28	17

TABLE 2
Quantitative tracking results on the **TIO dataset**. The best scores are marked in **bold**.

People-Car	Method	MODA ↑	FP ↓	FN ↓	IDS ↓
Seq.0	JDTracker	**0.71**	**0.02**	**0.03**	**0.01**
	TIF-MIP [44]	0.67	0.07	0.25	0.04
	KSP [2]	0.49	0.10	0.41	0.07
	LP2d [24]	0.47	0.05	0.48	0.06
	POM [15]	0.47	0.06	0.47	-
	SSP [34]	0.20	0.04	0.76	0.04
Seq.1	JDTracker	**0.65**	**0.15**	**0.11**	**0.01**
	TIF-MIP [44]	0.58	0.17	0.25	0.04
	KSP [2]	0.04	0.71	0.25	0.12
	LP2D [24]	0.02	0.77	0.21	0.17
	POM [15]	-0.21	0.98	0.23	-
	SSP [34]	0.00	0.75	0.25	0.12

TABLE 3
Quantitative tracking results on the **People-Car** dataset. The best scores are marked in **bold**.

improve the robustness of visual tracking systems when there are frequent object interactions.

For the **People-Car** dataset, we compare the proposed method with other 5 state-of-the-arts methods: successive shortest path algorithm (SSP) [34], K-Shortest Paths Algorithm (KSP) [2], Probability Occupancy Map (POM) [15], Linear Programming (LP2D) [24], and Tracklet-Based Intertwined Flows (TIF-MIP) [44]. The quantitative results are reported in Table 3. From the results, the proposed method obtains better performance than the baseline methods.

PPL-DA dataset We compare the proposed method to four popular trackers: Probabilistic Occupancy Map (POM) [14], K-Shortest Path (KSP) [3], and Hierarchical Trajectory Composition (HTC) [50]. We included the results of the recent neural network based method, MDNet [33], which employs a Multi-Domain Convolutional Neural Network for visual tracking. In MDNet, each object of interest (or domain) is represented as a separate CNN network. MDNet achieved state-of-the-art tracking performance in multiple visual tracking benchmarks [33]. We also include the tracker **ActionNet** proposed by Yoo et al. [53]. ActionNet can be considered as a variant of the proposed method since it employs reinforcement techniques as well and aims to learn to shift object bounding boxes in order to track objects. In contrast, our method additionally utilizes visibility statuses while training policy networks, and quantize action spaces in a continuous space rather than the discrete space used in [53].

Table 4 reports the quantitative results of various methods on PPL-DA dataset. Notably, our method can significantly reduce the number of ID switches (IDSW) on all scenarios, which is a critical indicator of the superiority of our method. Our method also outperforms ActionNet, which clearly demonstrates the superiority of the proposed policy-based jump-diffusion process. Fig. 4 visualizes exemplar results on this dataset. These four video sequences are challenging in varying ways, including low-resolution (top-left, top-right), distortions (bottom-left), occlusions (bottom-left, bottom-right) and illumination changes (all four scenes). Our method can robustly track these subjects over time.

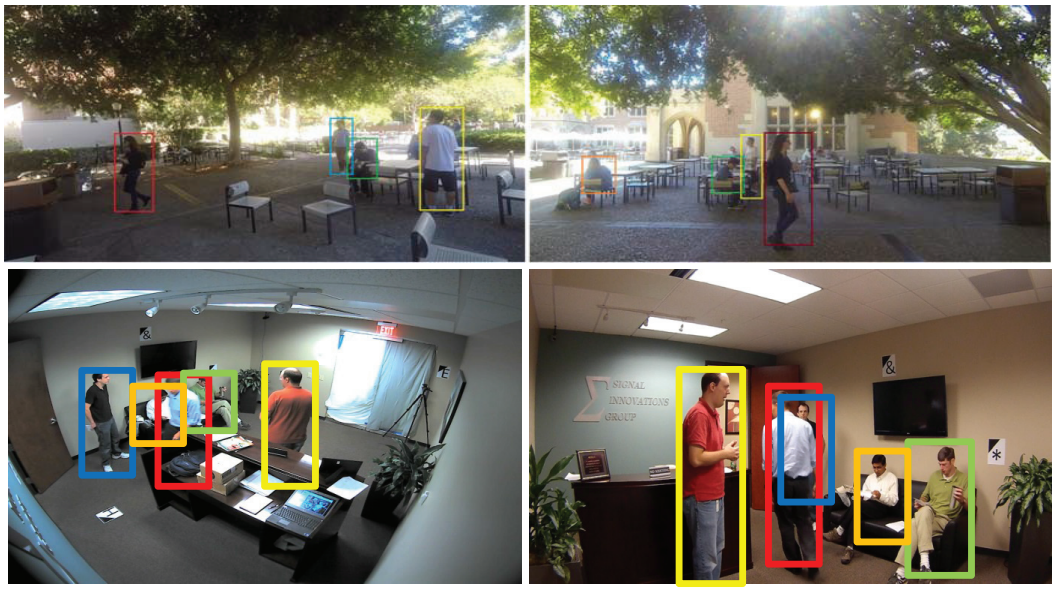

Fig. 4. Exemplar tracking results of the proposed method on the **PPL-DA** dataset.

Seq-Court	TA(%)	TP(%)	MT(%)	ML(%) ↓	IDSW ↓	FRG ↓
JDTracker	**54.8**	81.3	**35.7**	**17.2**	**18**	61
MDNet [33]	53.1	**82.1**	32.2	21.3	53	**42**
ActionNet [53]	45.6	78.1	28.7	26.8	61	51
HTC [50]	29.5	71.9	14.8	25.9	91	77
KSP [3]	24.7	64.4	0.00	44.4	318	291
POM [14]	22.3	65.4	0.00	51.9	296	269
Seq-Office	TA(%)	TP(%)	MT(%)	ML(%) ↓	IDSW ↓	FRG ↓
JDTracker	**66.5**	**89.5**	**64.3**	0.00	**18**	**25**
MDNet [33]	60.3	87.1	54.1	0.00	33	28
ActionNet [53]	57.3	78.5	55.3	0.00	34	32
HTC [50]	41.2	70.7	28.6	0.00	66	59
KSP [3]	39.6	58.0	28.6	0.00	83	76
POM [14]	36.9	58.8	28.6	0.00	89	82
Seq-Plaza	TA(%)	TP(%)	MT(%)	ML(%) ↓	IDSW ↓	FRG ↓
JDTracker	**48.9**	**76.5**	**43.4**	**11.0**	**59**	**72**
MDNet [33]	27.4	68.9	18.5	12.7	112	98
ActionNet [53]	32.5	65.8	29.5	21.5	99	103
HTC [50]	23.1	66.2	11.6	18.6	202	178
KSP [3]	17.3	57.5	7.0	27.9	356	311
POM [14]	16.7	57.9	4.6	32.6	339	295

TABLE 4
Quantitative tracking results on the **PPL-DA** dataset that includes three video sequences.

5 Conclusions and Future Works

This paper revisited the classical jump-diffusion process and studied a novel way to make it computationally feasible for dealing with real-time problems. We focus on visual tracking and study in particular how to jointly track objects and reason their visibility statues in videos. Our method employs two different dynamics to reconfigure the present solution in both discrete and continuous space, and chooses the optimal dynamic at each step through discriminaively trained policy functions. Integrating with the proposed Deep Reinforcement Learning methods, we developed an effective algorithm to learn policy functions from training data. Results on multiple public video benchmarks showed that our method can accurately recognize object visibility statuses and robustly track objects in crowded scenes.

The proposed method employs an alternative strategy to jointly optimize the continuous and discrete policy networks. The outcomes of the continuous policy network will be used to move or scale the target box and thus change the present object state. The learning of discrete policy network is dependent on the new object state. However, the learning of continuous policy network is not dependent on the outcomes of discrete states. This unidirectional coupling relationship is reasonable since the deep policy representation can work well without explicitly knowing of visibility status. It might also help reduce the negative impact of errors made for visibility estimation, which is not ignorable at the early stage of joint learning.

The developed techniques can be applied to solve a variety of computer vision tasks, e.g., image reconstruction, segmentation, and activity recognition, which will be studied in the future. Taking image reconstruction for an instance, it often involves two

coupled sub-tasks: segmenting input images into semantic regions, and estimating the 3D coordinates of each semantic region, which are defined in continuous and discrete spaces, respectively. These two subtasks can be jointly solved by the proposed continuous-discrete inference algorithm. Similarly, the inference of image segmentation is traditionally formulated as a Markov Chain Process [56] and be solved by a stochastic jump-diffusion process, where jump dynamics are used to flip the colors of image segments and diffusion dynamics are used to refine segment boundaries. In contrast to the traditional stochastic policy making strategy, the proposed policy-based proposal making method can be used to accelerate the mixing of the Markov Chain. Therefore, the developed techniques provide a generic framework for joint inference in heterogeneous spaces, which has wide application potentials in computer vision.

Moreover, we plan to study the proposed reinforcement tracker from three aspects. The first research question is how tho effectively specify rewards for continuous/discrete actions in reinforcement learning. In this work, we determine rewards based on states only, regardless of actions. In continuous space, for example, it is possible to associate rewards with properties of the action sequences in order to encourage effective and compact actions. The second question is how to train policy functions with a limited amount of supervisions. Complete object trajectory annotations are very expensive to prepare. It is invaluable to investigate a weakly supervised way to train the proposed policy networks. The third question is the theoretical analysis of the proposed policy-based jump-diffusion process, which can help understand the underlying convergence properties.

Acknowledgment

Xiaobai Liu was supported by the DARPA SIMPLEX program (No. 58723A), National Science Foundation (No. 1657600), ONR grant (No. N00014-17-1-2867) and San Diego State University Presidential Leadership Funds.

References

[1] A. Andriyenko, K. Schindler, and S. Roth. Discrete-continuous optimization for multi-target tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[2] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *IEEE transactions on pattern analysis and machine intelligence*, 33(9):1806–1819, 2011.

[3] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua. multiple object tracking using k-shortest paths optimization. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819, 2011.

[4] J. C. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2488–2496, 2015.

[5] X. Chang, Y.-L. Yu, Y. Yang, and E. P. Xing. Semantic pooling for complex event analysis in untrimmed videos. *IEEE transactions on pattern analysis and machine intelligence*, 39(8):1617–1632, 2017.

[6] J. Choi, J. Kwon, and K. M. Lee. Visual tracking by reinforced decision making. *arXiv preprint arXiv:1702.06291*, 2017.

[7] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. In *Proceedings of the IEEE Conference on Computer Vision*, 2017.

[8] A. Dehghan, S. Assari, and M. Shah. Gmmcp-tracker:globally optimal generalized maximum multi clique problem for multiple object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[9] A. Dehghan, Y. Tian, P. Torr, and M. Shah. Target identity-aware network flow for online multiple target tracking. In *IEEE Conference on Computer Vision and Pattern Recogntion*, 2015.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[11] A. Ess, K. Schindler, B. Leibe, and L. V. Gool. Improved multi-person tracking with active occlusion handling. In *IEEE ICRA Workshop on People Detection and Tracking*, 2009.

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

[13] J. Ferryman and A. Shahrokni. Pets2009: Dataset and challenge. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009.

[14] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multi-camera people tracking with a probabilistic occupancy map. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 30(2):267–282, 2008.

[15] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multicamera people tracking with a probabilistic occupancy map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):267–282, 2008.

[16] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[17] P. J. Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[18] U. Grenander and M. I. Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 549–603, 1994.

[19] S. Hamid Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid. Joint probabilistic data association revisited. In *IEEE International Conference on Computer Vision*, 2015.

[20] F. Han, Z. Tu, and S.-C. Zhu. Range image segmentation by an effective jump-diffusion method. *IEEE Transactions on pattern analysis and machine intelligence*, 26(9):1138–1153, 2004.

[21] C. Huang, S. Lucey, and D. Ramanan. Learning policies for adaptive tracking with deep feature cascades. In *Proceedings of the IEEE Conference on Computer Vision*, 2017.

[22] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and J. Zhang. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):319–336, 2009.

[23] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited. In *IEEE International Conference on Computer Vision*, 2015.

[24] L. Leal-Taixé, M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese. Learning an image-based motion context for multiple people tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[25] B. Li, T. Wu, and S.-C. Zhu. Integrating context and occlusion for car detection by hierarchical and-or model. In *European Conference on Computer Vision*, 2014.

[26] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan. Perceptual generative adversarial networks for small object detection. In *IEEE CVPR*, 2017.

[27] J. Li, Y. Wei, X. Liang, J. Dong, T. Xu, J. Feng, and S. Yan. Attentive contexts for object detection. *IEEE Transactions on Multimedia*, 19(5):944–954, 2017.

[28] Z. Ma, X. Chang, Y. Yang, N. Sebe, and A. G. Hauptmann. The many shades of negativity. *IEEE Transactions on Multimedia*, 19(7):1558–1568, 2017.

[29] A. Maksai, X. Wang, and P. Fua. What players do with the ball: A physically constrained interaction modeling. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[30] A. Milan, K. Schindler, and S. Roth. Multi-target tracking by discrete-continuous energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2054–2068, 2016.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[32] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[33] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4293–4302, 2016.

[34] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[35] H. Possegger, T. Mauthner, P. Roth, and H. Bischof. Occlusion geodesics for online multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[36] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Conference on Neural Information Processing Systems*, 2015.

[37] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[38] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah. Part-based multiple-person tracking with partial occlusion handling. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[39] J. Supancic III and D. Ramanan. Tracking as online decision-making: Learning a policy from streaming videos with reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[40] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[41] S. Tang, M. Andriluka, and B. Schiele. Detection and tracking of occluded people. *International Journal of Computer Vision*, 110(1):58–69, 2014.

[42] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 101(1):184–204, 2013.

[43] X. Wang, E. Turetken, F. Fleuret, and P. Fua. Tracking interacting objects optimally using integer programming. In *European Conference on Computer Vision*, 2014.

[44] X. Wang, E. Turetken, F. Fleuret, and P. Fua. Tracking interacting objects using intertwined flows. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 38(11):2312–2326, 2016.

[45] L. Wen, W. Li, J. Yan, and Z. Lei. Multiple target tracking based on undirected hierarchical relation hypergraph. In *IEEE Conference on Computer Vision and Pattern Recogntion*, 2014.

[46] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[47] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015.

[48] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *IEEE International Conference on Computer Vision*, 2015.

[49] Y. Xu, L. Lin, W.-S. Zheng, and X. Liu. Human re-identification by matching compositional template with cluster sampling. In *proceedings of the IEEE International Conference on Computer Vision*, pages 3152–3159, 2013.

[50] Y. Xu, X. Liu, Y. Liu, and S. Zhu. Multi-view people tracking via hierarchical trajectory composition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[51] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[52] A. Yilmaz, L. Xin, and M. Shah. Contour based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 26(11):1532–1536, 2004.

[53] S. Yoo, J. Yun, Y. Choi, K. Yun, and J. Y. Choi. Action-decision networks for visual tracking with deep reinforcement learning. 2017.

[54] S.-I. Yu, D. Meng, W. Zuo, and A. Hauptmann. The solution path algorithm for identity-aware multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recogntion*, 2016.

[55] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. In *IEEE International Conference on Computer Vision*, 2015.

[56] S.-C. Zhu. Stochastic jump-diffusion process for computing medial axes in markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11):1158–1169, 1999.

Xiaobai Liu is an Assistant Professor of Computer Science at San Diego State University (SDSU), San Diego, U.S.A. He received his PhD from the Huazhong University of Science and Technology (HUST), China. His research interests focus on the development of theories, algorithms, and models for the core computer vision problems. He has published 45 peer-reviewed articles in top-tier conferences (e.g. ICCV, CVPR, etc.) and leading journals (e.g. TPAMI, TIP etc.). He received a number of awards for his academic contributions, including the outstanding thesis award by CCF (China Computer Federation).

Qian Xu is the co-founder and president of XreLab Inc., San Diego, CA. She received the Master degree and Doctoral degree from the Department of Statistics, San Diego State University in 2011 and 2017, respectively, and received the B.S. degree from the School of Science, Beihang University, Beijing, China, in 2006. Her research interest falls in the various statistical models and their applications in computer vision.

Thuan Chau is a Master student in the Department of Computer Science, San Diego State University, and currently works as a research assistant. His research interest falls in the areas of computer vision and machine learning.

Lei Zhu is a professor with the School of Information Science and Engineering, Shandong Normal University. He received his B.S. degree (2009) from Wuhan University of Technology, and the Ph.D. degree (2015) from Huazhong University of Science and Technology. He was a Postdoctoral Research Fellow at Data & Knowledge Engineering research group in The University of Queensland (2016 to 2017), and Singapore Management University (2015 to 2016). His research interests are in areas of multimedia analysis and search.

Yadong Mu received the Ph.D. degree from Peking University in 2009. He is now an assistant professor at Peking University and leading the machine intelligence lab at Institute of Computer Science and Technology. Before joining Peking University, he has ever worked at National University of Singapore, Columbia University, Huawei Noah's Ark Lab and AT&T Labs. His research interest is in large-scale machine learning, video analysis and computer vision.

Shuicheng Yan is chief scientist of Qihoo/360 company, and also the Dean's Chair Associate Professor at National University of Singapore. Dr. Yan's research areas include machine learning, computer vision and multimedia, and he has authored/co-authored hundreds of technical papers over a wide range of research topics, with Google Scholar citation over 20,000 times and H-index 66. He is ISI Highly-cited Researcher of 2014, 2015 and 2016. His team received 7 times winner or honorable-mention prizes in PASCAL VOC and ILSVRC competitions, along with more than 10 times best (student) paper prizes. He is a IEEE Fellow and IAPR Fellow.