One global optimization method in network flow model for multiple object tracking

Zhenyu He a, Yuxin Cui a,b, Hongpeng Wang a,c, Xinge You d,⇑, C.L. Philip Chen d,e

a Department of Computer Science, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China
b Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29201, USA
c Shenzhen IOT Key Technology and Application Systems Integration Engineering Laboratory, Shenzhen 518055, China
d Department of Electronics and Information Engineering, Huazhong University of Science and Technology, China
e The Faculty of Science and Technology, University of Macau, Macau

A R T I C L E   I N F O

Article history:
Received 9 November 2014
Received in revised form 14 March 2015
Accepted 19 April 2015
Available online 30 April 2015

Keywords:
Multiple object tracking
Network flow model
Observation model

A B S T R A C T

In this paper, we address the task of automatically tracking a variable number of objects in the scene of a monocular and uncalibrated camera. We propose a global optimization method in network flow model for multiple object tracking. This approach extends recent work which formulates the tracking-by-detection into a maximum-a posteriori (MAP) data association problem. We redefine the observation likelihood and the affinity between observations to handle long term occlusions. Moreover, an improved greedy algorithm is designed to solve min-cost flow, reducing the amount of ID switches apparently. Furthermore, a linear hypothesis method is proposed to fill up the gaps in the trajectories. The experiment results demonstrate that our method is effective and efficient, and outperforms the state-of-the-art approaches on several benchmark datasets.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Multiple object tracking is an important aspect of computer vision, especially the pedestrian tracking. It has been used in many tasks, such as the video surveillance and automatic drive. Unlike the single object tracking, which only keeps an eye on one object along the frame sequence, multiple object tracking must track all the targets in camera sight and address many complex situations, e.g., object enters and exits, one is occluded by another or some barriers, and so on. In single object tracking, we may just concentrate on how to represent the object which needs to be tracked. However, in multiple object tracking, the focus has been transformed to how to address the data association problem, i.e., how to find the corresponding observation in previous frames or next frames. Moreover, the multiple object tracker has to automatically track targets of a certain category, so that when a target emerges in the scene, the tracker should start a new tracking. If the camera is uncalibrated and monocular, lots of information will be lost when the 3D world is mapped to 2D image, which brings about some intricate occlusions. Different methods have been proposed to address the problem of tracking multiple objects using monocular and uncalibrated camera in recent years. Nevertheless, the factors such as the complicated background, the crowded people and the low-quality video can render the multiple tracking extremely challenging.

With the improvement of the detector [1–4] in recent years, tracking-by-detection [5–7] is becoming feasible and popular. In the framework of tracking-by-detection, it is not indispensable to observe the targets during the tracking procedure any more and we can concentrate on addressing the data association problem because high reliable object detections are given by the detector as the input observations. That is, we only need to connect the target hypotheses generated by the detector as their similarity across frames. Several models based on graph theory have been introduced to simulate the tracking problem, such as the Maximum Weight Independent Set (MWIS) model [6], Generalized Minimum Clique Graphs (GMCP) model [8] and network flow model [9–11]. Actually, the detection result may have many false positives and missing detections. The missing detections are mainly caused by occlusions. Some state-of-the-art detectors [12,13] have a good performance on the partial occlusion. But for the full occlusion, it is helpless. Hence, it is necessary for tracking-by-detection tracker to be able to eliminate the false positive and fill up the missing detections (see Fig. 1).

Tracking in network flow model is a kind of global optimal procedure. In this framework, the object responses, i.e., the nodes,
compose a large and intricate network graph. Two nodes are connected by an arc means that they may represent the same target and the cost of arcs or edges in the graph indicates the agreement between two detection observation. Pirsiavash et al. [5] formulated data association to an MAP problem and solved it as a min-cost flow issue. Rather than applying the proven optimal algorithms directly, such as the push relabel algorithm [14] and successive shortest path [15], they proposed a greedy method on the basis of finding the shortest path with dynamic programming algorithm. Their approach can return all the trajectories in extremely short time, not giving the global optimal solution but a high-quality approximate solution. Our work is mainly inspired by the min-cost flow network described in [5]. We find out that, in our cost flow network model, this greedy algorithm is not high-quality any more. We aim to reformulate the similarity evaluation to adapt the network flow mode to long term occlusions and improve Pirsiavash’s approximate greedy method to be not only efficient but also precise. The main contributions of this paper include:

- A new integrated observation model for evaluating the affinity between two detection observations. This integrated observation model not only has better robustness than previous models, but also can deal with the occlusions situation.
- An improved approximate greedy algorithm. For the problem that Pirsiavash’s approximate greedy may cause a lot of ID switches, this improved approximate greedy algorithm can remarkably eliminate the ID switches and its running time is much less than optimal algorithms.
- A simple and effective linear hypothesis method for reducing the number of false negatives. After the optimization, a lot of gaps caused by the occlusion exist in the trajectories though the detections have been connected to trajectories. This linear hypothesis mechanism can fill up most of the gaps accurately.

In the rest of this paper, we briefly discuss the related work in Section 2 and describe the construction of network flow model in Section 3. A novel optimization algorithm is proposed in Section 4 and a detailed experimental evaluation of the presented method is given in Section 5. Finally, a conclusion is drawn in Section 6.

2. Related work

Considerable developments have occurred for multiple targets tracking since radar tracking [16] twenty years before. Early approaches follow objects in local strategy [17,18]. That is, they solve the data association frame-by-frame and object-by-object. Once one object is found in one frame, the tracker will keep on looking for it in the next frame based on its state estimate in one or more previous frames. In this framework, kalman filter [19] and particle filter [20] which is known as a sequential Monte Carlo method, are usually applied to connect the object hypotheses.

Many approaches in global strategy are proposed in recent years. These methods usually address the global data association by solving the global optimization problem over a long period window whose range may be from tens of frames to more than one thousand frames. Contrasted to the approaches in local strategy, these approaches have a higher accuracy whereas the temporal delay is inevitable. Besides, since most of globally optimal methods employ the reliable detection output of a high-quality detector as their input, the performance of a globally optimal method can be heavily influenced by the input detection responses. According to their specific models, tracking task can be modeled to dynamic programming problem [21], cost-flow network [22,23], Maximum Weight Independent Set [6], Generalized Minimum Clique Graphs [8], and so on.

In the network flow model, the basic and first procedure is transforming the global data association to a maximum-a
posteriori (MAP) problem, and this MAP estimation can be derived to an objective function with some constrains which represent the real world. Then this objective function can be solved by an appropriate solver. There are various physical interpretations for each component of the network graph. In the network flow proposed by Berczal et al. [24], one node indicates a space–time location and each location can only contain one detection. The flows which connect the nodes mean the trajectories. The data association problem is formulated to a convex linear programming problem which can be solved by a standard linear programming solver. In [9], Berczal et al. formulated this linear programming problem as a $k$-shortest paths problem which is similar to min-cost flow problem to reduce the computational complexity. In both of these methods, the transition from one location to another only considers the locations’ position, which can induce ID switches when two objects are too close. Later in [25], Berczal et al. added the appearance information to the network flow model and reformulated the tracking problem as an optimization of a convex objective function to address the ID switch issue.

The locations in [24,9,25] are rectangular and the ground plane is divided by chessboard grid. In [11], the ground plane is discretized by hexagonal lattice instead of the chessboard to make smoother and more accurate trajectories. In [22], the global data association problem is formulated as a classic and standard cost-flow network model. Each node (except the start and sink node) means an observation of a detection response, including an object’s position, scale, appearance, and time step, and indicates a space–time location without the appearance in [5]. Nevertheless, the network flow models in [5] is the same as that in [22]. Push-relabel method is employed in [22] to solve the min-cost flow problem with an extremely high computational complexity. The successive shortest-path algorithm, a typical algorithm to solve min-cost flow problem, for tracking is proposed in [5] to significantly reduce the computational complexity from $O(n^m \log n)$ to $O(Kn \log n)$, where $K$ is the number of the trajectories, $n$ and $m$ are the number of node and edges in the network model, respectively. Moreover, an approximate greedy algorithm which can find the min-cost flows in $O(Kn)$ is proposed in [5]. This algorithm can guarantee a high detection accuracy but has a poor performance on dealing with the ID switches.

Based on [22,5], we modify the network flow model to make it be able to handle long-term occlusions and propose an improvement on Pursiavash’s greedy algorithm to get one more optimal solution for min-cost flow network. By comparing with the method in [5] and several recent state-of-the-art methods, we show the efficacy of the proposed work.

3. Network flow model

According to [22], we model the tracking problem as an MAP estimation problem. Then, an objective function can be derived from this MAP problem by representing the data association as a Hidden Markov Model (HMM), and figured as a min-cost flow [15,22,5] problem.

3.1. MAP estimate

Let the detection observation box, yielded by object detector, be denoted by $x_i = \{p, a, s, m, t, \text{con}_i\}$, where $p$ is the spatiotemporal position, $a$ is the appearance, $s$ is the box size, $m$ is the motion, $t$ is the time step and $\text{con}_i$, provided by detector, is the confidence of the object observation. The set of all object observations along the frame sequence can be represented by $X = \{x_i\}$, and one trajectory can be written as $T_k = \{x_k, x_{k+1}, x_{k+2}, \ldots, x_{k+n}\}$, which is a list of detection observation ordered by incremental frame number. The tracking hypothesis $T$ is the set of trajectories $T = \{T_k\}$. Then, the global data association problem can be formulated as an MAP problem [22,5], as shown in Eq. (1).

$$
T^* = \arg\max_{T} P(T|X) = \arg\max_{T} \prod_{i} P(x_i|T) P(T)
$$

with

$$
P(x_i|T) = \begin{cases} 
P(\text{con}_i|T) \quad \exists T_k \in T, x_i \in T_k \\ P(\text{con}_i|fp) \quad \text{otherwise} 
\end{cases}
$$

where $P(x_i|T)$ is the likelihood of observation $x_i$ respect to $T$, and $fp$ means the false positive detection. $P(\text{con}_i|T)$ and $P(\text{con}_i|fp)$ are both normal distribution, but quite different in our experiment.

Intuitively, all of trajectories are independent with each other because no intersection would occur among them and each trajectory can be modeled as a Markov chain. Thus, $P(T)$ in Eq. (1) can be transformed as:

$$
P(T) = \prod_{i \in T} P(T_k)
$$

with two constraints:

$$
f_u f_d f_g f_f \in \{0, 1\} 
$$

$$
f_u + \sum_j f_j = f_i = f_u + \sum_j f_j 
$$

where

$$
c_i = -\log P_i(x_i) \quad c_u = -\log P_i(x_0) 
$$

$$
c_y = -\log P_i(x_i) \quad c_d = -\log P_i(x_{t+1}) 
$$

$c_u, c_d, c_i, c_y$ are the cost of observation $i$ being the start of track and being the end of track, the transition from $i$ to $j$, and observation likelihoods, respectively. $f_u$ is 1 if and only if observation $x_i$ can be linked to $x_j$, $f_d$ is 1 means that $x_i$ is the start or end of a track, respectively. And $f_f$ is 1 when $x_i$ is in a trajectory. The second constraint makes sure that one observation only belongs to no more than one track, so that, all the tracks are independent. We refer the reader to [22] for the derivation from Eqs. (1)–(4). The network flow model for three consecutive frames is shown as Fig. 2, which is similar to the network flow illustrated in [22,5]. The difference between them is that the transition linking between two non-consecutive frames is shown explicitly. In [22], the start and end cost $c_u$ and $c_d$ are estimated by the number of trajectories and number of detection hypotheses. In addition, all the $c_u$ are defined the same as $c_u$. But actually, in most cases, one object must first appear at the edge of scene when it comes into the scene of a camera, and it must also disappear at the edge rather than anywhere when it goes away. So the probability of the track starts or ends at the edge of the scene is significantly larger than the probability of the track starts or ends at the center area of the scene. In a certain scene, it is easy to find out the entrance and exit area, as shown in Fig. 3.

Then, the track start probability $P_i(x_i)$ can be defined as:
distinguish the different targets. The probability of transition from observation $x_i$ to $x_j$ can be defined as:

$$P(x_j|x_i) = (\varphi(\Delta t)P(a_i,a_j,\Delta t) + \omega(\Delta t)(P(p_j|p_i,\Delta t)P(m_j|m_i,\Delta t)))P(s_j|s_i,\Delta t)$$  \hspace{1cm} (8)$$

where $\Delta t$ is the time gap between the detection observation $i$ and $j$. $\varphi(\Delta t)$ is the coefficient of appearance transition possibility and $\omega(\Delta t)$ is the coefficient of position & motion similarity. 

$$P(a_i|a_j,\Delta t) = \frac{P_i}{P_i + P_j} \quad \text{if } a_i \neq a_j$$

$$P(a_i|a_i,\Delta t) = \frac{P_i}{P_i + P_j} \quad \text{if } a_i = a_j$$

$P(a_i|a_j,\Delta t)$ is the transition cost which indicates the number of trajectories whose head or tail is far from the objects and the poses for the same object may be distant from each other. We only use the color information for the appearance model. Let $a_i$ be a 36-bin HSV histogram extracted from detection observation $x_i$. The appearance similarity $\varphi(\Delta t)$ is a simple monotonous rising linear function when $\Delta t$ is less than a threshold $\theta$, which is obtained according to the framerate of specific frame sequences. $\varphi(\Delta t)$ is defined as:

$$\varphi(\Delta t) = \begin{cases} 
0.5 & \Delta t = 1 \\
0.5 + 0.25(\Delta t - 1)/\theta & 1 < \Delta t < \theta \\
0.75 & \theta \leq \Delta t \leq \Theta 
\end{cases}$$

where $\Theta$ indicates the maximal allowed time gap. $P(a_i|a_j,\Delta t)$ represents the appearance similarity. Since the textural features of different people may resemble each other especially when the camera is far from the objects and the poses for the same object may be distinct from each other, we only use the color information for the appearance model. Let $a_i$ be a 36-bin HSV histogram extracted from detection observation $x_i$. We denote the appearance distance between two detection responses as $D_{ij}$, which is obtained according to the framerate of specific frame sequences. $D_{ij}$ is the Bhattacharyya distance between the two histograms. Figs. 4 and 5 show the distribution of the distance between two different objects and between two same objects with various $\Delta t$. We find out that this distribution is similar to the typical Poisson distribution and the distribution between two same objects and the distribution between two different objects have a statistically significant difference. Since the stochastic variables in Poisson distribution must be integer values, the $D_{ij}$ is multiplied to be in an appropriate range and discreted by the approximate integer. Let $TD_{ij}$ be the discreted distance. Then, $P(a_i|a_j,\Delta t)$ can be represented as:

$$P(a_i|a_j,\Delta t) = \frac{\text{Pois}(TD_{ij}; \lambda_{i,j,M})}{\text{Pois}(TD_{ij}; \lambda_{i,j,M}) + \text{Pois}(TD_{ij}; \lambda_{i,M})}$$

$\lambda_{i,j,M}$ is the number of detections and $\lambda_{i,M}$ can be defined as:

$$\lambda_{i,M} = \sum_{t} \sum_{j} \lambda_{i,j,M}$$

$\lambda_{i,j,M}$ represents the number of detections $i$ and $j$ are in two consecutive frames, the number of trajectory $k$ such that $k\leq \lambda_{i,j,M}$.

The track end probability $P_t(x_i)$ is defined as:

$$P_t(x_i) = \begin{cases} 
K_x & P_i \in R \\
0 & \text{otherwise} 
\end{cases}$$

where $R$ is the exit/entrance area (covered by red oblique lines in Fig. 3). $K_x$ means the number of trajectories whose head or tail is in the exit/entrance area, $N_x$ represents the number of detections in $R$. $C$ is the place at which one object unlikely appears at the first time or disappears permanently, i.e., the area except $R$. $K_c$ indicates the number of trajectories except $K_x$. $N_c$ is the number of detections in $C$, and $p_i$ is the position of detection $i$. The track end probability $P_t(x_i)$ is defined the same as $P_t(x_i)$. The parameters $K_x$ and $K_c$ are estimated according to the ground truth and $N_x$ is estimated by the detection input as well as $N_c$.

### 3.2 Observation model

For multiple tracking task, handling the connection between objects is the key. The transition cost indicates which target one observation most likely belongs to. And the inaccuracy of transition cost estimation would cause a lot of ID switches. Due to the complicated condition (people wear similar clothes, walk in the same direction or are very close to each other), only one feature cannot effectively distinguish the targets in the real scene. We measure the similarity between two hypotheses by fusing several intrinsic properties, such as the color, size, spatiotemporal position and motion. This integrated feature can well represent and
where $\Delta t$ is the mean of discreted distance between two detection
observations, which represent the same target, with a time gap $\Delta t$
between them. $\Delta t_\text{dis}$ indicates the mean of the discreted distances
between two different objects. In our experiments, both of $TD_{ij}$
between the same objects and different objects can be modeled as
Poisson distribution. Considering the time gap between two observations,
we compute $TD_{ij}$ and get the Poisson distributions within $\Delta t$ in the range of 1 to $\Theta$ (distinct by the frame rate of datasets).

$$P(p_i|p_j, \Delta t) = \exp(-x||p_j - (p_i + m_i\Delta t)||)^2)$$

(11)

where $\alpha$ is a scale parameter, $m_i$ indicates the movement velocity of
observation $x_i$, which can be generated by the optical flow. Different
parts of human body may have different movement information.
For instance, one leg sometimes may be static and one arm may
swing backward when a person is walking forward. Hence, we
select the body part (about 1/4–1/2 from head top) to represent
the whole person. $P(m_i|m_i)$ denotes the similarity of movement
and can be written as:

$$P(m_i|m_i) = \exp(-\sigma||m_i - m_i||)^2).$$

(12)

where $\sigma$ is a scale parameter. Further, the similarity of detection
size is represented as:

$$P(s_j|s_j, p_j, p_j) = \min\left(\frac{s_j - \beta(h_j - h_j)}{s_j}, \frac{s_j - \beta(h_j - h_j)}{s_j}\right)$$

(13)

where $h_j$ means the y coordinate value of the object’s position on
image, and $s_j - \beta(h_j - h_j)$ indicates the size that observation $x_j$
should be if it is at position $p_j$. $\beta$ can be precisely computed by
aldata when the camera is calibrated, as in PETS datasets. If not, we
can give it an approximate value and estimate it based on
ground truth.

4. Optimization

4.1. Approximate greedy programming algorithm

Pirsiavash et al. [5] proposed a novel approximate greedy algo-
rithm based on Dynamic Programming (DP) to find min-cost flows.
Contrasted to the classic push-relabel method [14] with computa-
tional complexity $O(n^2m \log n)$ or Successive Shortest-Paths (SSP)
algorithm [15] with computational complexity $O(Kn \log n)$, this
greedy algorithm decreases the computational complexity to
$O(Kn)$, where $K$ is the number of found trajectories, $n$ is the number
of nodes and $m$ is the number of edges in the network graph.

In the first step of the approximate greedy algorithm in [5], an
$O(n)$ dynamic programming algorithm is employed to compute the
shortest cost path of each node from the start node s in the initial
graph. The set of observations which can be transited to i, i.e. link
to node i, is denoted as $N(i)$ and the min-cost for observation i is
cost(i). Before updating the cost(i), all the min-costs are initialized
to cost(i) = $c_i + c_{\text{su}}$. Then the cost(i) will be updated recursively by:

$$\text{cost}(i) = c_i + \min(\pi, c_{\text{su}})$$

(14)

where $\pi = \min_{j \in N(i)} (c_j + \text{cost}(j)).$ Hence, the min-cost node can be
found, and the shortest path from s to t can be generated by a back-
ward procedure starting from the min-cost node. The second step of
the approximate greedy algorithm is removing the nodes on the
path from graph and updating the cost(i) in the sub-graph. By iter-
ating those two steps until no path whose cost is lower than the
shortest threshold can be found, the final solution will be generated.
Theoretically, this approximate greedy algorithm makes it impos-
sible to modify the found paths in subsequent procedure, so it cannot
produce the global optimal solution. In the experiment of [5], this
approximate greedy algorithm can yield high-quality approximate
solutions because all the transition costs $c_{\text{su}}$ have been set at the
same value zero. Virtually, the similarity between two observations
must be different and the probability of transition is almost impos-
sible equal to 1. Hence, if the transition costs $c_{\text{su}}$ are not equal, this
approximate greedy algorithm may cause lots of unacceptable ID
switches, as shown in Fig. 6.

4.2. Improved greedy programming algorithm

We propose some measures to deal with the problem caused by
approximate greedy algorithm and return more optimal solution
for our tracking problem. Our essential purpose is to ensure the
shortest path, found and removed from graph, does not need to
be modified any more. Hence, we just modify the shortest path
before it is deleted.

Our motivation is that among the neighborhoods of an observa-
tion, the one which belongs to the same trajectory with that
and recompute the cost(i) which may change because of the disconnection. Then, we find the shortest path literally and modify it until every node on the path is the optimal choice for its previous node.

It must be noticed that above procedure may cause some problems. For example, it may split the path by the reason of little patches which are composed by only few observations (nodes) in short frame sequences. Actually, the patch usually is false positive path and should be ignored. If it is added into T, one new redundant trajectory will be produced and ID switch would happen, as shown in Fig. 7. An additional process is required to handle it. When one node has better choices to connect than the next node on the path, we need to determine whether all of the optional successive nodes belong to a patch. If it does, we will not modify the shortest path, but ignore these optional successive nodes. Otherwise, we do the process of cutting the link on the path. To judge whether one node i belongs to a patch, we search from i until finding a node, whose cost of the path from i to itself, is less than a threshold or no more nodes which are connected to i can be searched.

Although our algorithm cannot be proven as a global optimal algorithm, in practice, we find this algorithm already has an optimal performance in most test frame sequences. Inevitably, our algorithm would bring more iterations than the original approximate greedy algorithm. Nevertheless, the compute complexity will not rapidly increase if the graph is appropriate. In our algorithm, we can roughly consider that the additional iterations is caused by potential ID switches. Once one more iteration happens, one possible ID switch has been eliminated. So the number of iterations is mainly decided by the complexity of tracking, such as how often does the occlusion occur, not the number of targets and the length of the sequences. In our experiments, the number of iterations usually scales linear with K and the scale is apparently less than log n. The cost of each iteration is equal to the greedy algorithm in [5] which is O(nK). Thus, the total computer complexity scales linear with O(Kn). So, our algorithm still costs much less time than SSP to generate a high-quality tracking result.

Algorithm 1. Improved greedy algorithm based on DP

```
Initial the network graph while no negative cost path can be found do
    Find the shortest path $T_k$ in the graph using DP.
    $cut = 0$
    for i from $length(T_k)$ to 1 do
        if $cut == 1$ then break end if
        if $T_k$ is the one which is most similar to node $T_{k-1}$ in $N(T_{k-1})$ then
            for each node(y) $\in N(T_{k-1})$& node(y) $\neq T_k$ do
                if node(y) is not in a path then
                    Cutting the link between $T_k$ and $T_{k-1}$
                    $cut = 1$
                    break end if
            end for
        end if
    end for
end while
```
be the observation after the gap and \( m \). \( x \) and \( y \) are the x axis value and y axis value of \( m \). However, the shortage is that it cannot speculate the missing detections if the object’s motion has changed acutely during the occlusion. Due to this case, if the time gap is larger than a threshold, it will not be filled up to avoid the situation after the gap, the hypothesis can fill up the gaps by others or the environment. That means the tracker must have the ability to hypothesize the possible positions of the occluded objects.

5. Evaluation

5.1. Datasets

In this section, we show the performance of our method on two popular public datasets: PETS2009 [26] and TUD datasets [27]. In PETS2009, we select the PETS2009-S2L1-view001 as the testing sequence. TUD datasets contains three frame sequences: TUD crossing (201 frames), TUD campus (71 frames) and TUD stadtmitte (179 frames). Both of these two datasets are very challenging and have distinct characteristics. PETS09 sequences are low frame rate, the same target in PETS09 between consecutive frames may have significant shift whereas this situation would not happen in TUD sequences. Andriyenko et al. have annotated all targets in each frame no matter whether the targets are occluded by others or the environment. That means the tracker must have the ability to hypothesize the possible positions of the occluded objects which are impossibly found by any detectors.

5.2. Metrics

The widely used CLEAR MOT metrics [28] is employed to evaluate our system. It is composed by two intuitive metrics: MOTA and MOTP. MOTA (Multiple Object Tracking Accuracy) measures the false positives, false negatives and ID Switches. In the tracking-by-detection framework, because we just use the result of detector as our input data, this metric almost comprehensively assesses the performance of the tracker. Additionally, we also evaluate the false positive ratio (Precision), false negative ratio (Recall) and the mount of ID Switch, respectively, to show more details. MOTP(Multiple Object Tracking Precision) only shows the ability of the tracker to give the object exact corresponding hypothesis. The larger the distance between the hypothesis and the object is, the less MOTP is. Our work depends on the output of the detector. To a certain extent, MOTP just estimates the performance of the detector we employ and we do not pay much attention to improve it by correcting the detection’s position.

5.3. Implementation

In our experiments, we employ the state-of-the-art deformable part model detector [2] to generate the target hypotheses which may still include lots of false negatives and false positives in each frame. In PETS09 frame sequence, the objects are too small to be detected. Hence, we scale the original frame to double size to make the object with size about 30 \( \times \) 60. The detection threshold is 0 for datasets is normal frame rate (25 fps), low definition and shot at the horizontal angle. The most occlusions in PETS09 are less than 10 frames while they can be very long in TUD. Moreover, because of the low frame rate, the same target in PETS09 between consecutive frames may have significant shift whereas this situation would not happen in TUD sequences. Andriyenko et al. have already given the ground truth on their website.\(^1\) They annotated all targets in each frame no matter whether the targets are occluded by others or the environment. That means the tracker must have the ability to hypothesize the possible positions of the occluded objects which are impossibly found by any detectors.

![Fig. 7. Illustration of the problem caused by splitting path. In this graph, the value of \( c_s \) and \( c_t \) is set to 1 and each \( c_i \) is set to 2. In the first iteration, the shortest path (s-a-b-c-h-t) is found. And if we cut the link \( k_i \), in the end, we will get two paths (s-a-b-f-t) and (s-c-h-t). The node f can be seen as a path.](image)

![Fig. 8. Linear hypothesis method. Each hollow circle means a detection response in our optimization result. At time \( t \), the detector cannot detect the object, so there is a false negative between \( t-1 \) and \( t+1 \). The motion and position have not acutely changed during this gap. In this situation, linear hypothesis can give a reasonable hypothesis for the missing detection (The red circle is the hypothesis). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)](image)

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Prec</th>
<th>Rec</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETS</td>
<td>92.6</td>
<td>81.7</td>
</tr>
<tr>
<td>Crossing</td>
<td>99.4</td>
<td>98.2</td>
</tr>
<tr>
<td>Stadtmitte</td>
<td>83.3</td>
<td>67.4</td>
</tr>
<tr>
<td></td>
<td>96.1</td>
<td>90.1</td>
</tr>
<tr>
<td></td>
<td>80.0</td>
<td>60.8</td>
</tr>
<tr>
<td></td>
<td>97.7</td>
<td>78.9</td>
</tr>
</tbody>
</table>

\(^1\) http://www.gris.informatik.tu-darmstadt.de/aandriye/data.html.
PETSO9 dataset, and is −1 for TUD dataset because of its low frame quality. The scale parameters in position and motion similarity are set to \( a = 0.01 \) and \( \sigma = 0.1 \). The maximal allowed time gap \( \Theta \) and threshold \( \theta \) are set to \( \Theta = 25 \) and \( \theta = 10 \) for PETSO9 sequences and \( \Theta = 30 \) and \( \theta = 15 \) for TUD sequences. That means our method can handle the occlusion less than 25 frames in PETSO9 and less than 30 frames in TUD sequences, respectively. Larger maximal allowed time gap means more computation. In practice, our setting is

![Frame 269](image1)
![Frame 271](image2)
![Frame 274](image3)

**Fig. 9.** A illustration of linear hypothesis which is used to eliminate the false negative. The first row shows the raw optimization result. The person whose ID is 1 in frame 269 is missing in frame 271 and reappear in frame 274. The second row is the output processed by the linear hypothesis, where this person is still labeled as ID 1 in frame 271.

![Frame 10](image4)
![Frame 11](image5)

**Fig. 10.** The first row is the tracking result in PETS sequence yielded by the observation model in [5]. The two frames are two consecutive frames, frame 10 and frame 11. The person whose ID is 108 and the person whose ID is 238 in frame 10 change their ID to 25 and 231 in frame 11 respectively. The second row is the tracking result generated by our integrated observation model, where this ID switch does not occur.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>MOTA</th>
<th>MOTP</th>
<th>IDX</th>
<th>Prec</th>
<th>Rec</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETS</td>
<td>80.7</td>
<td>77.9</td>
<td>381</td>
<td>99.2</td>
<td>89.6</td>
</tr>
<tr>
<td></td>
<td><strong>88.7</strong></td>
<td><strong>77.9</strong></td>
<td><strong>29</strong></td>
<td><strong>99.7</strong></td>
<td><strong>89.6</strong></td>
</tr>
<tr>
<td>Crossing</td>
<td>66.8</td>
<td>78.8</td>
<td>29</td>
<td>99.4</td>
<td>69.9</td>
</tr>
<tr>
<td></td>
<td><strong>78.0</strong></td>
<td><strong>78.9</strong></td>
<td><strong>8</strong></td>
<td><strong>96.8</strong></td>
<td><strong>81.4</strong></td>
</tr>
</tbody>
</table>

**Table 2** Comparison of CLEAR MOT [28] results on typical sequences using original greedy optimization algorithm.
enough to handle most occlusions. Moreover, the threshold for node linking in the network flow graph is 0.5.

5.4. Tracking performance

Although using the state-of-the-art detector, a mass of false positives and false negatives still exist in the detection output. Despite the main purpose of tracking is to address the data association problem, the tracker also has the responsibility to remove the false positives and the false negatives. In our cost flow model, the cost of the detection response which has higher confidence is lower, so the false positives may have higher cost than others. Through the optimization process, most of false positives are removed and at least about half of false positives can be filled by the linear hypothesis (see Table 1). In Table 1, the first line of each frame sequence shows the evaluation of raw detection result generated by the deformable part model detector [29] and the second line shows the results after tracking processing. Fig. 9 shows an example of filling the missing detection by linear hypothesis.

To evaluate the performance of our observation model, including the $P_t(x_t), P_t(x_t), P(x_t|x_0)$ and $P(x_t|T)$, we compare it with the model in [5] in several benchmark datasets. Table 2 illustrates the comparison result. The first line of each frame sequence shows the results generated according to the observation model in [5], and the second line shows the results yielded by our integrated observation model. We employ the approximate greedy optimization algorithm in [5] as the solver. The code of [5] is provided by Pirsiavash on his website. From this result, we can see that our observation model is more robust and accurate. The number of ID switch is more than three hundred for PETS sequence, and it is just 29 for TUD crossing sequence. That is because the framerate in PETS2009-S2L1-View001 is lower than normal rate and object’s position may apparently change even in consecutive frames, so that only the position and size state cannot represent the objects very well and one track can be often segmented to a few tracklets (see Fig. 10). Although the framerate of crossing sequence is normal, when the tracks of two objects intersect, ID switch still often occurs (see Fig. 11). Our method integrates the object’s appearance, position, size and motion together, and thus has a great adaptability no matter the framerate is low or high.

Besides comparing our observation model with the model in [5], we also compare our optimization algorithm with the original approximate greedy algorithm in [5]. Table 3 shows the comparison of these two optimization algorithms tested on the PETS2009-S2L1-View001 sequences and TUD crossing sequence. All of these experiments share the same network graph and the only difference is the optimization procedure. The first line of each frame sequence shows the optimization result of original approximate greedy algorithm, the second line shows the optimization result of our improved greedy algorithm without handling the

![Frame 81](image1.png)  ![Frame 93](image2.png)

**Fig. 11.** The first row shows the trajectories generated by the observation model in [5] in TUD crossing sequence. In frame 81, the person whose ID is 3 will be occluded by the person whose ID is 6 in the next frames, and thus his ID is changed to 15 in frame 93. The second row is the tracking result generated by our integrated observation model, where this ID switch does not occur.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MOTA</th>
<th>MOTP</th>
<th>IDS</th>
<th>Prec</th>
<th>Rec</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETS [5]</td>
<td>88.7</td>
<td>77.9</td>
<td>29</td>
<td>99.7</td>
<td>89.6</td>
<td>-9454.1</td>
</tr>
<tr>
<td>Crossing</td>
<td>78.0</td>
<td>78.9</td>
<td>8</td>
<td>96.8</td>
<td>81.4</td>
<td>-1343.1</td>
</tr>
<tr>
<td></td>
<td>85.4</td>
<td>78.4</td>
<td>5</td>
<td>95.6</td>
<td>90.1</td>
<td>-1702.2</td>
</tr>
<tr>
<td></td>
<td>86.3</td>
<td>78.4</td>
<td>2</td>
<td>96.1</td>
<td>90.1</td>
<td>-1781.5</td>
</tr>
</tbody>
</table>

Table 3

Comparison of CLEAR MOT [28] results about the optimization algorithms on typical sequences.

---

2 http://www.ics.uci.edu/dramanan.
patches, and the third line shows the result optimized by our whole improved greedy algorithm. From this comparison result, we can see that our algorithm largely decreases the amount of ID Switch from tens to few (also as shown in Figs. 12, 13 and 14). The flow cost for PETS2009-S2L1-View001 made by [5] is $9454.1$ while our algorithm can achieve $11,612$. And for TUD crossing, the cost generated by the greedy algorithm [5] is $1343.1$ while our algorithm can give the solution whose cost is $1781.5$. Apparently, our optimization algorithm outperforms the original approximate greedy algorithm in [5].
Moreover, we compare our experiment result with several state-of-the-art methods. These methods are not only limited in network flow framework but also implemented by many feasible and successive approaches, including Generalized Minimum Clique Graphs (GMCP) tracker [8], discrete–continuous optimization [30], particle filter based on tracking-by-detection [7], continuous energy minimization in [27] and Maximum Weight Independent Set (MWIS) model [6]. The comparison results are shown in Table 4. TUD contains three sequences (Campus, Crossing and Stadtmitte). We give the average performance of these sequences because they have similar characteristics, and are significantly different from PETS. In terms of MOTA, our method achieves the best performance on all of these datasets.

Moreover, we compare our experiment result with several state-of-the-art methods. These methods are not only limited in network flow framework but also implemented by many feasible and successive approaches, including Generalized Minimum Clique Graphs (GMCP) tracker [8], discrete–continuous optimization [30], particle filter based on tracking-by-detection [7], continuous energy minimization in [27] and Maximum Weight Independent Set (MWIS) model [6]. The comparison results are shown in Table 4. TUD contains three sequences (Campus, Crossing and Stadtmitte). We give the average performance of these sequences because they have similar characteristics, and are significantly different from PETS. In terms of MOTA, our method achieves the best performance on all of these datasets.

The amount of ID switches is apparently decreased in PETS. For the TUD Crossing sequence (201 frames), our matlab optimization code can find the solution in less than 3 s. Hence, our method satisfies the real-time requirement. Our method use a global strategy to achieve a good performance on tracking. It means we need to collect the observations on all of the frames to the memory. So the video duration would be limited by the memory size. In practice, it would be less than one hour. If the video duration is too long to process, we may need to use the sliding window strategy to handle it.

6. Conclusion

In this paper, we have presented an integrated observation model to measure the similarity between two observations and an improved greedy algorithm for solving min-cost flow problem in the typical network flow model. Our integrated observation model can precisely describe the characteristics of object observations produced by detectors and formulate the affinities between them. Hence, even a person is occluded by more than tens of frames, once he/she is detected again, our tracker can find out the link between his/her two observations. For the optimization procedure, our optimization method improved upon greedy programming algorithm is able to yield a satisfactory tracking result in much less time than that of simultaneously processing. Finally, when the trajectories have been found, a linear hypothesis method is employed to fill up the gaps in the trajectories to make them smooth and complete. Moreover, there are still some aspects that we need to improve. First, we only allow the time gap for less than 30 frames. If one object is occluded for more than the threshold, this method cannot handle it. Seconds, the movement similarity evaluation is based on the assumption that the object has a constant speed. If the speed changes rapidly in very short time, this method would fail.

Acknowledgements

This study was supported by the Innovation Project of Scholars from Overseas of Shenzhen (KQCX20120801104656658) and the Technology Innovation Project of Shenzhen (Nos. CXXZZ20120618155717337, CXXZZ20130318162826126). This
research was also supported in part by Shenzhen IOT key technology and application systems integration engineering laboratory. The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

References


