Enhancing Linear Programming with Motion Modeling for Multi-target Tracking

Niall McLaughlin Jesus Martinez Del Rincon Paul Miller
Centre for Secure Information Technologies (CSIT), Queen's University Belfast
nmclaughlin02@qub.ac.uk

Abstract

In this paper we extend the minimum-cost network flow approach to multi-target tracking, by incorporating a motion model, allowing the tracker to better cope with long-term occlusions and missed detections. In our new method, the tracking problem is solved iteratively: Firstly, an initial tracking solution is found without the help of motion information. Given this initial set of tracklets, the motion at each detection is estimated, and used to refine the tracking solution. Finally, special edges are added to the tracking graph, allowing a further revised tracking solution to be found, where distant tracklets may be linked based on motion similarity. Our system has been tested on the PETS S2.L1 and Oxford town-center sequences, outperforming the baseline system, and achieving results comparable with the current state of the art.

1. Introduction

Multi-target tracking in video consists of detecting all subjects in every frame, and following their complete trajectory over time. Successful research on a new generation of reliable pedestrian detectors [5, 11] has prompted the use of the tracking-by-detection paradigm [6], even for crowded or semi-crowded scenarios. Under this paradigm, tracking can be understood as a two-step process composed of detection and data association. The tracker first acquires a set of detections using a pedestrian detector. The individual detections are then assigned to tracks, where each track is composed of all the detections from a single individual. If all persons were to be correctly observed at every time-step this task would be trivial, however, due to false positive detections, occlusions and missed detections, this association problem becomes very challenging. In this paper we explore the potential of linear programming (LP), using a minimum-cost network flow formulation, for solving the multi-target tracking problem. Specifically, we investigate techniques for improving the data association of detections into tracks by using motion information.

The minimum-cost network flow tracking approach can cope well with short sequences of missed detections, however it tends to become unreliable given relatively long periods of occlusion or detector failure. This problem occurs because the association costs, used to link detections into tracks, are based only on static features such as appearance similarity or distance between detections. These static features tend to be unreliable indicators of whether detections belong to the same track if there are large gaps between the detections.

To address this problem, we propose to solve the tracking problem iteratively. Initially, static features, such as distance and time, are used to find a preliminary tracking solution given the set of detections over a window of frames. Using these initial tracklets, the motion at each detection is estimated and used to remove associations incompatible with a linear velocity assumption. We then introduce the concept of tracklet linking to the minimum-cost network flow tracker, by adding special edges to the tracking graph, allowing widely separated tracklets to be linked based on motion similarity. Association of detections, and association of tracklets, makes use of a novel cost function shown to outperform a Gaussian type cost function on these tasks.

1.1. Related Work

Recent research has demonstrated the potential of global optimization for tracking in crowded scenes, where the matching problem is solved jointly for all tracks [4, 18, 21]. Under this perspective, multi-target tracking is defined as a constrained min-cost flow optimization problem, that can be solved using LP to find the global optimum. As an additional advantage, trajectory initialization and termination are inherent to this methodology.

Given a set of pedestrian detections, it is possible to model the problem of the multi-target pedestrian tracking using graph theory. Typically in such an approaches, the vertexes of the graph may be used to represent the discrete locations where pedestrians are permitted to exist [3, 12], or pedestrian locations as hypothesized by a detector [4, 16, 8].

Graph edges are typically used to model the cost of associating pairs of detections in the same track, based on features such as appearance or distance [8]. Due to the fact that problems in graph theory can be expressed as equivalent linear programs, an assortment of LP tracking models have been explored including: k-shortest paths [4], flow linear programming [3] and min-cost network flow [8]. Such methods are appealing due to their mathematically rigorous formulation, but may require special vertexes to represent occlusions [21, 15], missed detections [20], or higher-order motion constraints [16], leading to increased model complexity.

In LP approaches to tracking, global optimization is carried out over a sequence of frames using static features such as appearance-similarity or distance between detections. In contrast with predictive tracking approaches, such as Kalman filtering [13] or Particle Filtering [6], motion is typically not included during this optimization, leading to difficulties when dealing with long-term occlusion. Only a few papers have considered the inclusion of motion models or constraints in the LP framework to deal with long term occlusions. An approach where global optimization is combined with social behavior analysis to improve accuracy in crowded scenarios is presented in [16]. Their model takes into account past and future frames to include constant velocity and social interaction. However, this potential is not fully exploited since only instant constant velocity is assumed and the social forces could be impractical or counter-productive in highly crowded scenes [15]. Similarly, a smoothness constraints over a three frame windows, based on instant constant velocity, is introduced by [8] and [9]. Finally, hypergraphs [17, 10] have been proposed to allow higher order motion models in the edges of a normal graph. Although this makes it possible to handle an arbitrary higher-order cost functions over the entire trajectory, it has not been shown to favorably impact the tracking performance or to help during long term occlusions [8].

2. Multiple Person Tracking

2.1. Multi-target Tracking as a Data-Association Problem

A pedestrian detector can be used to produce a set of detections over a sequence of frames, the goal of multi-target tracking is to correctly associate all the detections corresponding to each individual. Let the set of all detections, of size I, be $D = \{d_{i_1} \dots d_i \dots d_I\}$. Each detection is defined as $d_i = (x, y, w, h, c, t)$ where (x, y) is the position of the detection in image space, (w, h) are the bounding box width and height, c is the detector confidence, and t is the time associated with the detection. A tracklet is an ordered set of detections $\tau = \{d_{k_1} \dots d_k \dots d_K\}$, of size K, where $d_k(t) < d_{k+1}(t) \, \forall k$, and $[k_1 \dots K] \subset [1, I]$. The goal of multi-target tracking is to find the set of N tracklets $T* = \{\tau_1 ... \tau_N\}$ which best explains the detections i.e. the set of tracklets with the maximum posterior probability given the detection set: $T* = \arg\max_T P(T|D)$. As direct search for the optimal set of tracklets would be intractable, a common simplifying assumption [16] is that every detection may belong only to a single tracklet, and that detections are conditionally independent, which can be expressed as

$$T* = \arg\max_T \prod_i P(d_i|T)P(T) \tag{1}$$

$$T* = \arg\max_T \prod_i P(d_i|T) \prod_n P(\tau_n) \tag{2}$$

where $P(d_i|T)$ is the probability that d_i is a true detection, and $P(\tau_n) = P(\{d_{k,1} \dots d_k \dots d_K\})$ reflects the plausibility of a given tracklet τ_n, which can be expressed, using the Markovian assumption as follows

$$P(\tau_n) = P_{\mathcal{S}}(d_{k,1}) P_{\mathcal{E}}(d_K) \prod_k P_{\mathcal{L}}(d_k|d_{k-1}) \tag{3}$$

where $P_{\mathcal{S}}(d_{k,1})$ is the probability the tracklet starts at $d_{k,1}$, $P_{\mathcal{E}}(d_K)$ is the probability the tracklet ends at d_K, and $P_{\mathcal{L}}(d_k|d_{k-1})$ is the probability that detection d_{k-1} follows detection d_k within tracklet τ_n.

2.2. Minimum-Cost Network Flow Problem

The above generally defined multi-target tracking problem can be modeled as a minimum-cost network flow problem, where detections, and the cost of associating two detections, are modeled as graph edges with associated costs. Given a directed graph $G = (V, E)$, with special vertexes: source, $s \in V$, and sink, $r \in V$. Every edge, $E_{(u,v)}$, has an associated cost, $c_{(u,v)} \in \mathcal{R}$, flow capacity constraint, $u_{(u,v)} \geq 0$, and flow, $f_{(u,v)} \geq 0$. The goal is find the set of edges $Q \subseteq E$, capable of transporting X units of flow from S to T, with minimum cost, without violating the capacity constraints, where the cost for a given solution, Q, is defined as $\sum_{(u',v') \in Q} C_{(u',v')} F_{(u',v')}$.

The objective function in Eq. (2) can be translated into an equivalent minimum-cost network flow problem by first taking the logarithm of Eq. (2) and then incorporating Eq. (3), which leads to the following linear function

$$\begin{aligned} T* = \text{argmin}_T - \sum_i \log P(d_i) - \sum_n (\log P_{\mathcal{S}}(d_{k_1}) \\ - \log P_{\mathcal{E}}(d_K) - \sum_k \log P_{\mathcal{L}}(d_k|d_{k-1})) \end{aligned} \tag{4}$$

This function can then be mapped to the minimum-cost network flow framework by incorporation of flags $f_{i,j} \in \{0, 1\}$

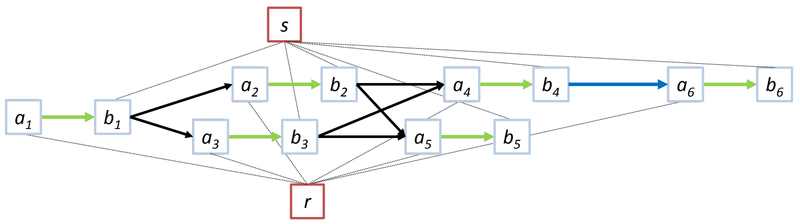

Figure 1. Graph used to represent the tracking problem. Every green edge, (a_i, b_i), represents a detection d_i. Black edges, (b_i, a_j), are standard links between detections. **The blue edge** (b_i, a_j) **represents our proposed link between widely separated detections**. Vertexes s and r are the source and sink vertexes. Gray edges, (s, b_i) and (a_i, r) connect the source and sink vertexes to all detections.

for the flow along edge (i, j), and costs $C_{i,j} \in \mathcal{R}$ indicating the cost associated with traversing edge (i, j), as follows

$$T* = \text{argmin}_T \sum_i C_i f_i + \sum_{i,j} C_{i,j} f_{i,j} + \sum_i C_{i,r} f_{i,r} + \sum_i C_{s,i} f_{s,i} \tag{5}$$

where calculation of the costs associated with each edge type: $C_i \propto P(d_i)$, $C_{i,j} \propto P_{\mathcal{L}}(d_k|d_{k-1})$, $C_{s,i} \propto P_{\mathcal{S}}(d_{k_1})$ and $C_{i,r} \propto P_{\mathcal{E}}(d_K)$, will be discussed in later sections. This expression of the tracking problem allows linear programming to be used to efficiently compute a minimum-cost network flow solution. To ensure the solution to the minimum-cost network flow problem is also a valid tracking solution, several constraint are placed on the flag variables. As detections may only be a member of a single tracklet, Eq. (1), a capacity constraint, $u_i \leq 1, \forall u_i$, is placed on all edges, and flags variables are subject to the constraints

$$f_{s,i} + f_i \leq 1 \qquad f_{i,r} + f_i \leq 1 \tag{6}$$

which enforce mutual exclusion at the starting and ending detections of each tracklet. The following additional constraints enforce conservation of flow at each detection

$$f_i + f_{s,i} = \sum_j f_{i,j} \qquad f_i + f_{i,r} = \sum_j f_{j,i} \tag{7}$$

The above described minimum-cost network flow problem can be used to construct a graph G, describing the relationships between all detections, as follows: Every detection d_i is represented by an edge connecting two vertexes (a_i, b_i) with associated cost C_i and flag f_i, where the cost depends on the detector confidence. All detections are connected to the source vertex s, by an edge (s, b_i) with cost $C_{s,i}$ and flag $f_{s,i}$, and are connected to the sink vertex r, by an edge (a_i, r) with cost $C_{i,r}$ and flag $f_{i,r}$. The relationships between all pairs of detections in frames $d_i(t)$ and $d_j(t')$, where $t' > t$, are represented as linking edges (b_i, a_j), with cost $C_{i,j}$ and flag $f_{i,j}$. An illustration of the tracking graph for a simple scenario is shown in Fig. 1.

2.2.1 Detection Linking Edges

A pair of detections, d_i and d_j, are connected by edge (b_i, a_j) with cost $C_{i,j}$, shown as black edges in Fig. 1. Detections may be linked if the following conditions are met: The time gap $\Delta T_{i,j}$ between the detections must satisfy $\Delta T_{i,j} \leq \Delta T_{\max}$. The distance $D_{i,j}$ between the detections must satisfy $D_{i,j} \leq D_{max}$ where $D_{\max} = \Delta T_{i,j} V_{\max}$ i.e. the speed required to link the detections must be less than $V_{\max}$ units per frame. Finally, the overlap of the bounding box sizes defined as, $O_{i,j} = (d_i \cap d_j)/(d_i \cup d_j)$, must satisfy $O_{i,j} \geq 0.5$. If these conditions are satisfied the linking cost is defined as

$$C_{i,j} = \mathcal{C}(D_{i,j}, D_{\max}) + \mathcal{C}(\Delta T_{i,j} - 1, \Delta T_{max}) \tag{8}$$

where $\mathcal{C}$ is defined as

$$\mathcal{C}(\kappa, \lambda) = 1 - e^{-\sqrt{\frac{\kappa}{\lambda}}} \tag{9}$$

so that linking cost increases with both distance and time, while links between detections in the consecutive frames incur no time-penalty cost, thus encouraging short links.

2.2.2 Detection Edges

All detections d_i are represented by two vertexes connected by inner edge (a_i, b_i) with cost C_i, shown as green edges in Fig. 1. The trivial solution to the min-cost network flow problem is $f_{i,j} = 0, \forall f_{i,j}$, with total cost zero i.e. no tracklets. To prevent this, all detection edges have negative cost i.e. $C_i < 0, \forall d_i$. Therefore the minimum-cost solution may take negative values by inclusion of detection edges, thus encouraging the formation of tracklets. The cost of all detection edges is defined as $C_i = -P(d_i)$, where $P(d_i)$ is the likelihood that detection d_i is a true detection, based on the normalized detector confidence score, $0.5 \leq d_i(c) \leq 1$. Thus, confident detections will have more negative scores, encouraging their inclusion in tracklets.

2.2.3 Source and Sink Edges

All detections d_i are connected to the source vertex by an edge (s, b_i), and to the sink vertex by an edge (a_i, r), with respective costs, $C_{s,i}$ and $C_{i,r}$, shown as gray edges in Fig. 1. We adopt the approach of [16] whereby $C_{s,i} = 0, \forall C_{s,i}$ and $C_{i,r} = 0, \forall C_{i,r}$. Therefore, no penalty or bonus is given for beginning or ending a tracklet. Additionally, due to the flow constraints of Eq. (6) and Eq. (7), and the fact that the source node is connected to the end of every detection edge, while the sink node is connected to the start of every detection edge, there is no bonus or penalty for beginning or ending at any particular detection. As every detection is treated equally, tracklets may begin or end at any location within the tracking area.

3. Motion Modeling For Linear Programming

In the previous section we have described the general framework for multi-target tracking using linear programming. One drawback of this approach is that velocity is not taken into account when calculating the association cost between individual detections. Unlike trackers with an explicit motion model, such as Kalman filtering [13] or particle filtering [6], optimization of the minimum-cost network flow problem is carried out simultaneously over a window of frames using only static features. While this approach can cope with short sequences of missed detections, long periods of occlusions and detector failure may cause it to become unreliable. To allow tracking to continue under such circumstances, the concept of tracklet linking, using a velocity cost, is introduced into the minimum-cost network flow tracker. We propose an iterative strategy, whereby initial tracklets are estimated using only static features, and are then refined based on inferred tracklet velocity.

3.1. Velocity estimation

Given an initial set of tracklets T, calculated using only static features, an estimated velocity can be associated with every detection that is a member of a tracklet, and can be used to both refine the initial tracklets as well as to link widely separated tracklets.

For tracklet $\tau \in T$, consisting of an ordered set of detections $\tau = \{d_1 ... d_S\}$, tracklet velocity at each detection can be estimated using first order linear regression, based on the assumption that realistic motion is linear over short time-periods. This approach has an important advantage in comparison to simple estimation of the instantaneous velocity [16] of the tracklet given the connected detection: it is more robust against outliers and inaccuracies in the detection locations, which can cause huge miscalculations in the estimated velocity since no filtering or smoothing is inherent in this schema.

For every detection $d_i \in \tau$, its velocity is estimated using two sets of neighboring detections: set $w_f = \{d_j ... d_i\}$ of detections ending at d_i, and set $w_b = \{d_i ... d_k\}$ of detections starting at d_i, where $|d_l(t) - d_i(t)| \leq F, \forall d_l \in w_b, w_f$, where F is the frame-rate of the sequence i.e. velocity is estimated using a 1s window of frames. For sets w_b and w_f, independent linear regressions are performed over the x, y, w and h values of the member detections with time as the predictor variable. Detection d_i is then written as $d_i = (x, y, w, h, c, t, \beta^f, \alpha^f, \beta^b, \alpha^b)$, where $\beta^f = (x^f_\beta, y^f_\beta, w^f_\beta, h^f_\beta)$ and $\alpha^f = (x^f_\alpha, y^f_\alpha, w^f_\alpha, h^f_\alpha)$ are the regression coefficients learned from w_f. Coefficients β^b and α^b learned from w_b are defined similarly. Using the regression coefficients, the tracklet state predicted forward from d_i to time t' can be written as $(x', y', w', h', t') = d_i(\beta^f)t' + d_i(\alpha^f)$.

3.2. Tracklet Refinement

The estimated velocity at each detection is used to refine the tracklets by breaking links that are not consistent with a linear velocity assumption over short periods of time. For every detection, not at the start or end of a tracklet, its associated regression coefficients are used to predict the tracklet state forward and backward in time, to the time of its following and preceding detections respectively. The bounding box intersections of the predicted states with the following and preceding detections are computed. If either intersection is zero, the tracklet is broken at that point. This process removes cases where a tracklet appears to make very sudden changes in velocity, which are unlikely to occur in realistic situations due both to the generally high detection rate as well as physical constraints such as inertia.

3.3. Edge Costs for Linking Over Long Gaps

Using the refined tracklets and estimated motion models, tracklet linking over long gaps can be performed using a third iteration of the LP optimization. Edges (a_i, b_j), shown in blue in Fig. 1, representing potential links between tracklets, are added to the tracking graph, between detections at the starting and ending points of the tracklets estimated in the previous iteration. These new edges have costs based on the relative velocities of the tracklets linked, while the costs associated with all other edges remain identical to those described in Section 2.2.

Detections d_e at the end of a tracklet τ_i, and d_s at the start of tracklet τ_j, may be linked if the following conditions are met. The time-gap $\Delta T_{e,s}$ between the detections must satisfy $\Delta T_{e,s} \leq \Delta \widetilde{T}_{\max}$, where $\Delta \widetilde{T}_{\max} \gg \Delta T_{\max}$ i.e. this link would be prevented by the time constraint $\Delta T_{e,s} \leq \Delta T_{\max}$ of Section 2.2.1. An additional time constraint $\Delta T_{e,s} \leq \max(|\tau_i|, |\tau_j|)$, where $|\tau_i|$ and $|\tau_j|$ are the length of τ_i and τ_j respectively, prevents long links between short tracklets, thus reducing the number of false positives. The speed constraint is identical to that used in Section 2.2.1, while the detection overlap constraint from Section 2.2.1 is not used during tracklet linking. If all conditions are satisfied, the cost $C_{e,s}$ for linking d_e with d_s is calculated as follows.

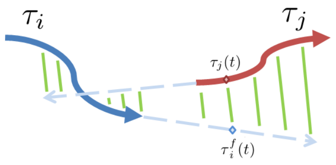

Figure 2. During tracklet linking, tracklet τ_i is projected forward in time, while tracklet τ_j is projected backward. The linking energy cost measures the average difference between the predicted and actual tracklet positions, e.g. between $\tau_i^f(t)$ and $\tau_j(t)$

Let $\tau_i^f(t)$ be the forward predicted (x, y) position of tracklet τ_i at time t, calculated using $d_e(\beta^f, \alpha^f)$, and let $\tau_j^b(t)$ be the backward predicted (x, y) position of tracklet τ_j at time t, calculated using $d_s(\beta^b, \alpha^b)$. Let $\tau_i(t)$ and $\tau_j(t)$ be the actual (x, y) positions of tracklets τ_i and τ_j respectively at time t, based on the member detections, and finally let $d_s(t)$ and $d_e(t)$ be the times associated with detections and d_s and d_e respectively. Then, a linking energy function can be defined in terms of the residual between the predicted and actual tracklet positions, as follows

$$\begin{aligned} E_{e,s} =& \frac{1}{F} \sum_{t'=1}^{F} |\tau_i^f(d_s(t) + t') - \tau_j(d_s(t) + t'))|_2 \\ &+ \frac{1}{F} \sum_{t'=1}^{F} |\tau_j^b(d_e(t) - t') - \tau_i(d_e(t) - t'))|_2 \end{aligned} \tag{10}$$

where F is the window length over which the energy function is calculated, typically equal to the frame-rate of the sequence. The linking energy function is illustrated in Fig. 2. The full tracklet linking cost $C_{e,s}$ is defined as

$$C_{e,s} = \mathcal{C}(E_{e,s}, E_{\max}) + \mathcal{C}(\Delta T_{e,s}, \Delta\widetilde{T}_{max}) \tag{11}$$

where the function $\mathcal{C}$ is defined in Section 2.2.1 and $E_{\max}$ is a large constant used to normalize the linking energy cost. The tracklet linking cost is designed to favor pairs of tracklets with a small residual between the predicted and actual positions of the tracklets. The linking energy cost $E_{e,s}$ is zero for tracklet pairs that lie in a straight line, meaning tracklet pairs consistent with the automatically estimated motion model are favored during linking. Once the linking costs have been added to the tracking graph, the minimum-cost network flow solution is computed again to produce the final tracking solution.

4. Experiments

The tracker was tested on the Oxford town center [2] and PETS S2.L1 [14] sequences. Example output is shown in Fig. 3. The Oxford dataset is 4500 frames, at 25 fps, and features realistic pedestrian motion in a moderately crowded street scene. The PETS S2.L1 sequence is 794 frames at 7 fps. Crowd density is low, however this sequence features more unpredictable pedestrian motion. For the Oxford sequence, pedestrian detection was performed offline using the part-based Poselets pedestrian detector [5]. For the PETS sequence the detections from [1] were used. As it has been shown that linear programming trackers can be sensitive to false positive detections [4], we pre-process the detector output as follows: Detections with a confidence score below a threshold are removed. Non-maxima suppression, with a threshold of 0.5, as in [11], is used. Finally, camera calibration is used to removed detections based on height, estimated using bounding-box size, for detections outside the range 1.2 m to 2.1 m.

The linear program representing the tracking problem is solved using the built-in Matlab solver `linprog`. As in [21] Fibonacci search is used to find $T*$, the optimal quantity of flow to pass through the tracking graph, where each unit of flow represents a tracklet. The final tracker output was smoothed using a moving average filter, with window length, F, equal to the frame-rate of the sequence. The parameters were set as follows: $D_{\max} = 20$ pixels/frame, $\Delta\widetilde{T}_{\max} = 3$ s, $E_{\max} = 200$, $\Delta T_{\max} = 3$ frames for PETS, and $\Delta T_{\max} = 5$ frames for Oxford.

4.1. Linking Parameters

In this experiment we investigate tracker performance when varying the tracklet linking method, and the cost function for associating detections and tracklets. We compare tracklet linking performance using the energy function proposed in Section 3.3, with a baseline method, and we compare the cost function proposed in Eq. (9), with a Gauss cost function. The baseline tracklet linking method simply extrapolates a line between the end detection of the first tracklet and start detection of the next tracklet, forward and backward by F frames. The association cost is then calculated similarly to Eq. (10). A Gauss cost function is defined as $c = 1 - e^{-(\frac{x}{x_{\max}})^2}$, where $x_{\max}$ is the maximum permitted value for x. This function grows slowly for small x, in contrast with our proposed cost function, which grows quickly for small x.

Tracking results for all combinations of tracklet linking method (baseline and regression) and cost function (Gauss and Eq. (9)) are shown in Fig. 4 as a function of $\Delta\widetilde{T}_{\max}$, the maximum gap in seconds, over which tracklet linking may

Figure 3. Example tracker output for, Left: Oxford Town Centre, Right: PETS S2.L1. Videos containing the tracking results are provided as supplementary material.

occur (see Section 3.3); $\Delta\widetilde{T}_{\max} = 0$ is equivalent the min-cost flow network with no long-distance links. From Fig. 4, it can again be seen that velocity and motion modeling improves performance, and that our cost function, Eq. (9), outperforms a conventional Gaussian cost function. For both sequences, as $\Delta\widetilde{T}_{\max}$ is increased, the combination of regression linking and proposed cost function produces the most stable performance over a wide range of $\Delta\widetilde{T}_{\max}$ values, with the desirable property of never decreasing performance below the system without a motion model, in spite of the parameter tuning. On the PETS sequence, the combined baseline methods are able to produce a higher MOTA for $\Delta\widetilde{T}_{\max} = 2$ however, this is not replicated on the Oxford sequence. This could be due to the short-term occlusions and random motion in the PETS sequence, meaning the motion model is less useful. Additionally, this combination is quite sensitive to parameter tuning. In contrast, when $\Delta\widetilde{T}_{\max}$ is increased to large values, meaning the tracker is permitted to link very widely separated detections, increasing the possibility of mistakes, tracking performance with the proposed methods does not drop below the level of no tracklet linking for PETS, and remains high for the Oxford sequence. In conclusion, these results show that the combination of our proposed tracklet linking method and cost function, can lead to improved tracker accuracy without requiring careful parameter tuning, as both sequences contain quite different tracking environments in terms of pedestrian motion, and different detectors were used to produce the detections for each sequence.

4.2. Tracking Results

Tracking results for the Oxford and PETS sequences are presented in Table 2 along with comparative results from the literature. Our results are comparable with the literature [14, 1], and our tracker out-performs other min-cost flow approaches, including [16] which uses simple instant velocity within LP. Comparing MOTA for the system tested with and without tracklet linking, we can see that motion modeling improves performance on both sequences, and although precision drops slightly due to some tracklets being incorrect linked, recall significantly improves, indicating that more tracklets were correctly linked. Additional statistics, shown in Table 1, were collected to compare use of stage 1 (Conventional LP), with the full system consisting of stages 1, 2 and 3 (Our Method), showing the number and length of tracking gaps present compared to the ground truth. For the complete system, on both datasets the number of gaps decreases across all lengths, with the number of short gaps (less than 0.5 s) decreasing significantly. The full system was also able to link tracklets across many medium length gaps, of 1 to 2 s, and can even successfully link across some very challenging long range gaps, of up to 75 frames or 3 s, in the case of the Oxford dataset.

Oxford	Gap Len. (s)	<0.5	<1	<2	≥ 2
	Gap Len. (Frames)	<4	<7	<14	≥ 21
	Our Method	28	10	6	4
	Conventional LP	46	21	10	4
PETS S2.L1	Gap Len. (s)	<0.5	<1	<2	≥ 3
	Gap Len. (Frames)	<12	<25	<50	≥ 75
	Our Method	478	167	95	48
	Conventional LP	647	237	116	52

Table 1. Tracker statistics with stage 1 only (Conventional LP), and with stages 1,2 and 3 (Our Method), shown as a histogram of the number of gaps present, up to a given gap-length.

5. Conclusion

In this paper we have proposed a novel way of incorporating motion modeling into a min-cost network flow tracker, by adding edges to the tracking graph to represent links between widely separated detections. The linking costs are based on a novel energy function that takes velocity into account. We show that this method can be used to complete large tracking gaps caused by occlusion or detector failure. The proposed method has shown good performance on the Oxford town-centre and PETS S2.L1 sequences, outperforming conventional LP methodologies and matching state of the art performances.

References

[1] A. Andriyenko and K. Schindler. Multi-target tracking by continuous energy minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1265–1272, June 2011.

[2] B. Benfold and I. Reid. Stable multi-target tracking in real-time surveillance video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3457–3464. IEEE, 2011.

[3] J. Berclaz, F. Fleuret, and P. Fua. Multiple object tracking using flow linear programming. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pages 1–8. IEEE, 2009.

[4] 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, Sept 2011.

[5] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *International Conference on Computer Vision (ICCV)*, 2009.

[6] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *International Conference on Computer Vision*, pages 1515–1522, Sept 2009.

[7] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Online multiperson tracking-by-detection from a single, uncalibrated camera. *IEEE Transactions on Pattern*

		MOTA	MOTP	Prec.	Recall
Oxford Town Centre	Our Method (Tracklet Linking)	71.89	72.94	89.53	81.41
	Conventional LP (No Tracklet Linking)	69.38	73.16	91.67	76.32
	H. Izadinia *et al.* [14]	75.7	71.6	93.6	81.8
	L. Leal-Taixe *et al.* [16]	67.3	71.5	71.6	67.6
	L. G. Shu *et al.* [19]	72.9	71.3	-	-
PETS S2.L1	Our Method (Tracklet Linking)	91.24	80.44	97.43	93.71
	Conventional LP (No Tracklet Linking)	88.44	80.70	98.32	89.98
	H. Izadinia *et al.* [14]	90.7	76.0	96.8	95.2
	L. Leal-Taixe *et al.* [16]	67.0	-	-	-
	A. Andriyenko *et al.* [1]	89.3	56.4	-	-
	M. Breitenstein *et al.* [7]	79.7	56.3	-	-

Table 2. Comparing tracker accuracy with Our Method (stages 1, 2 and 3, with $\Delta\widetilde{T}_{\max} = 3$ s), and Conventional LP (stage 1 only).

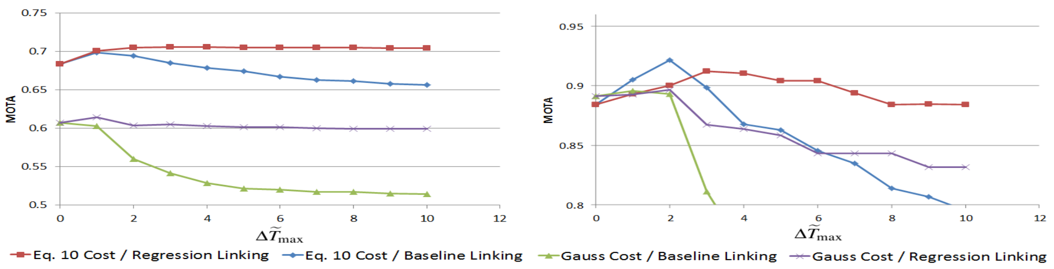

Figure 4. Tracker results in terms of MOTA, for differing tracklet linking methods and cost functions, as the parameter $\Delta\widetilde{T}_{\max}$ is varied. Left: Oxford Town Centre; Right: PETS S2.L1. Note, when $\Delta\widetilde{T}_{\max} = 0$ no tracklet linking is performed.

Analysis and Machine Intelligence, 33(9):1820–1833, Sept 2011.

[8] A. A. Butt and R. T. Collins. Multi-target tracking by lagrangian relaxation to min-cost network flow. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1846–1853. IEEE, 2013.

[9] A. A. Butt and R. T. Collins. Multiple target tracking using frame triplets. In *Computer Vision*, pages 163–176. Springer, 2013.

[10] R. T. Collins. Multitarget data association with higher-order motion models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1744–1751. IEEE, 2012.

[11] P. F. 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.

[12] 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.

[13] S. Iwase and H. Saito. Parallel tracking of all soccer players by integrating detected positions in multiple view images. In *International Conference on Pattern Recognition*, volume 4, pages 751–754 Vol.4, Aug 2004.

[14] H. Izadinia, I. Saleemi, W. Li, and M. Shah. 2t: multiple people multiple parts tracker. In *Computer Vision*, pages 100–114. Springer, 2012.

[15] H. Jiang, S. Fels, and J. J. Little. A linear programming approach for multiple object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[16] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In *IEEE International Conference on Computer Vision Workshops*, pages 120–127. IEEE, 2011.

[17] P. Ochs and T. Brox. Higher order motion models and spectral clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 614–621. IEEE, 2012.

[18] 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.

[19] 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*, pages 1815–1821, June 2012.

[20] B. Yang and R. Nevatia. Multi-target tracking by online learning of non-linear motion patterns and robust appearance models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1918–1925. IEEE, 2012.

[21] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.