Branch-and-price global optimization for multi-view multi-target tracking

Laura Leal-Taixé, Gerard Pons-Moll and Bodo Rosenhahn
Institute for Information Processing (TNT)
Leibniz University Hannover, Germany
{leal,pons,rosenhahn}@tnt.uni-hannover.de

Abstract

We present a new algorithm to jointly track multiple objects in multi-view images. While this has been typically addressed separately in the past, we tackle the problem as a single global optimization. We formulate this assignment problem as a min-cost problem by defining a graph structure that captures both temporal correlations between objects as well as spatial correlations enforced by the configuration of the cameras. This leads to a complex combinatorial optimization problem that we solve using Dantzig-Wolfe decomposition and branching. Our formulation allows us to solve the problem of reconstruction and tracking in a single step by taking all available evidence into account. In several experiments on multiple people tracking and 3D human pose tracking, we show our method outperforms state-of-the-art approaches.

1. Introduction

Combinatorial optimization arises in many computer vision problems such as feature correspondence, multi-view multiple object tracking, human pose estimation, segmentation, etc. In the case of multiple object tracking, object locations in the images are temporally correlated by the system dynamics and are geometrically constrained by the spatial configuration of the cameras (*i.e.* the same object seen in two different cameras satisfies the epipolar constraints).

These two sources of structure have been typically exploited separately by either Tracking-Reconstruction or Reconstruction-Tracking. Splitting the problem in two phases has, obviously, several disadvantages because the available evidence is not fully exploited. On the other hand, finding the joint optimal assignment is a hard combinatorial problem that is both difficult to formulate and difficult to optimize. In this paper, we argue that it is not necessary to separate the problem in two parts, and we present a novel formulation to perform 2D-3D assignments (reconstruction) and temporal assignments (tracking) in a single global optimization. When evidence is considered jointly, temporal correlation can potentially resolve reconstruction ambiguities and viceversa. The proposed graph structure contains a huge number of constraints, therefore, it can not be solved with typical Linear Programming (LP) solvers such as simplex. We rely on multi-commodity flow theory and use Dantzig-Wolfe decomposition and branching to solve the linear program.

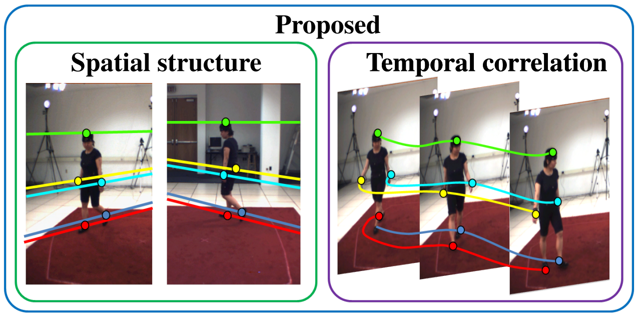


Figure 1: We propose to jointly exploit spatial and temporal structure to solve the multiple assignment problem across multiple cameras and multiple frames. With our proposed method both tracking and reconstruction are obtained as the solution of single optimization problem.

1.1. Related work

Multiple target tracking (MTT) is a key problem for many computer vision tasks, such as surveillance, animation or activity recognition. Occlusions and false detections are common, and although there have been substantial advances in the last years, MTT is still a challenging task. The problem is often divided in two steps: detection and data association. When dealing with multi-view data, data association is commonly split into two optimizations, namely *sparse stereo matching* and *tracking*. While stereo matching is needed for reconstruction (obtaining 3D positions from 2D calibrated cameras), tracking is needed to obtain trajectories across time. The tracking problem is usually solved on a frame-by-frame basis [10], using small batches of frames [12] or one track at a time [3]. Recent works show that global optimization using LP can be more reliable as it solves the matching problem jointly for all tracks. Specifically, the tracking problem is formulated as a maximum flow [4] or a minimum cost problem [8, 13, 15, 25], both efficiently solved using LP and with a far superior performance when compared to frame-by-frame or track-by-track methods. The sparse stereo matching problem for reconstruction is usually formulated as a linear assignment problem and it is well known that for more than 3 cameras the problem is NP-hard [16]. In [22] a comparison of the methods Tracking-Reconstruction vs. Reconstruction-Tracking is presented. In [4], first reconstruction is performed using Probabilistic Occupancy Map (POM), and then tracking is done globally using Linear Programming. In [24], the assignments are found using a data-driven MCMC approach, while [23] presented a formulation with two separate optimization problems: linking across-time is solved using networks flows and linking across-views is solved using set-cover techniques. In contrast to all previous works, we formulate the problem as a single optimization problem.

We propose a novel graph formulation that captures the whole structure of the problem which leads to a problem with a high number of constraints. This rules out standard Linear Programming solvers such as simplex [13, 25] or k-shortest paths [4, 15]. We define our problem as a multi-commodity flow problem, i.e., each object has its own graph with a unique source and sink. Multi-commodity flows are used in [19] in order to maintain global appearance constraints during multiple object tracking. However, the solution is found by applying several k-shortest paths steps to the whole problem, which would be extremely time consuming for our problem and lead to non-integer solutions.

By contrast, we use decomposition and branching methods, which take advantage of the structure of the problem to reduce computational time and obtain better bounds of the solution. Decomposition methods are closely related to Lagrangian Relaxation based methods such as Dual Decomposition [5, 11], which was used for feature matching in [21] and for monocular multiple people tracking with groups in [14]. In our case, we make use of the Dantzig-Wolfe decomposition [18] which allows us to take advantage the special block-angular structure of our problem. As is usual in multi-commodity flow problems, the solutions found are not integer and therefore branch-and-bound [6] is used. The combination of column generation and branch-and-bound methods is known as branch-and-price [2].

1.2. Contributions

In this paper, we propose a new global optimization formulation for multi-view multiple object tracking. We argue that it is not necessary to separate the problem into two parts, namely, reconstruction (finding the 2D-3D assignments) and tracking (finding the temporal assignments) and propose a new graph structure to solve the problem globally. To handle this huge integer problem, we introduce decomposition and branching methods which can be a powerful tool for a wide range of computer vision problems. To the best of our knowledge this is the first work to propose a global optimization scheme to perform multi-view as well as temporal matching for multiple target tracking. Finally, we make available a sample of the code used in this paper[1].

[1]http://www.tnt.uni-hannover.de/staff/leal/

2. Multi-view Multi-object tracking

Tracking multiple objects in several calibrated camera views can be expressed as an energy minimization problem. We define an energy function that at the 2D level (i) enforces temporal smoothness for each camera view (2D-2D), and at the 3D level (ii) penalizes inconsistent 2D-3D reconstructions from camera pairs, (iii) enforces coherent reconstructions from different camera pairs and (iiii) favors temporal smoothness of the putative 3D trajectories.

2.1. Proposed multi-layer graph

Matching between more than two cameras (k-partite matching) is an NP-hard problem. In order to be able to handle this problem, we propose to create a *multi-layer graph*. The first layer, the 2D layer, depicted in Figure 2(a), contains 2D detections (circular nodes) and the flow constraints and is where trajectories are matched across time. The second layer, the 3D layer, depicted in Figure 2(b), contains the putative 3D locations (square nodes) obtained from the 2D detections on each pair of cameras. It is designed as a cascade of prizes and favors consistent matching decisions across camera views. Thereby, the problem is fully defined as only one global optimization problem. In the following lines we define the types of edges of the proposed graph.

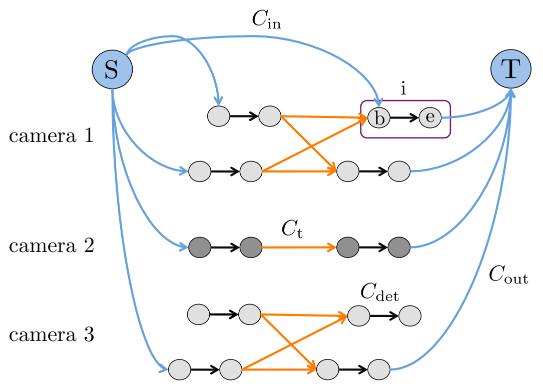
Entrance/exit edges ($C_{\text{in}}, C_{\text{out}}$). These edges determine when a trajectory starts and ends, the cost balances the length of the trajectories with the number of identity switches. Shown in blue in Figure 2(a).

Detection edges (C_{det}). To avoid the trivial zero flow solution, some costs have to be negative so that the solution has a total negative objective cost. Following [13, 25], each detection $\mathbf{p}_{i_v}$ in view $v \in \{1 \dots V\}$ is divided into two nodes, **b** and **e**, and a new detection edge is created with cost

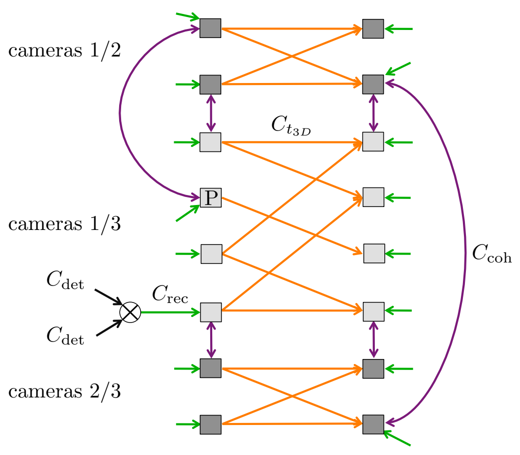
$$C_{\text{det}}(i_v) = \log\left(1 - P_{\text{det}}(\mathbf{p}_{i_v})\right).$$

The higher the likelihood of a detection $P_{\text{det}}(\mathbf{p}_{i_v})$ the more negative the cost of the detection edge (shown in black in Figure 2(a)), hence, confident detections are likely to be in the path of the flow in order to minimize the total cost.

Temporal 2D edges (C_{t}). The costs of these edges (shown in orange in Figure 2(a)) encode the temporal dynamics of



(a) 2D layer



(b) 3D layer

Figure 2: An example of the proposed multi-layer graph structure with three cameras and two frames.

the targets. Assuming temporal smoothness, we define $\mathcal{F}()$ to be a decreasing function [13] of the distance between detections in successive frames

$$C_{\text{t}}(i_v, j_v) = -\log\left(\mathcal{F}\left(\frac{\|\mathbf{p}_{j_v} - \mathbf{p}_{i_v}\|}{\Delta t}, V^{\text{2D}}_{\text{max}}\right) + B_f^{\Delta f - 1}\right),$$

where $V^{\text{2D}}_{\text{max}}$ is the maximum allowed speed in pixels and $B_f^{\Delta f - 1}$ is a bias that depends on the frame difference Δf and favors matching detections in consecutive frames. The function $\mathcal{F}$ maps a distance to a probability, which is then converted to a cost by the negative logarithm.

An energy function consisting of only the 2D layer is a special case of our multi-layer graph and would be suited to find the trajectories on each camera independently. To enhance the 2D tracking results with 3D information, we introduce the *3D layer* which contains three types of edges.

Reconstruction edges (C_{rec}). These edges connect the 2D layer (Figure 2(a)) with the 3D layer (Figure 2(b)). For each camera pair, all plausible 2D-2D matches create new 3D hypothesis nodes (marked by squares in Figure 2(b)). The reconstruction edges, shown in green, connect each newly created 3D detection with the 2D detections that have originated it. The cost of these edges encodes how well the 2D detections match in 3D which is implemented by computing the minimum distance between pairs of projection rays. Let $\mathcal{C}_v$ be the set of all possible camera pairs and m_k a new 3D hypothesis node generated from the 2D nodes i_{v_1} and j_{v_2}, where $k \in \mathcal{C}_v$ and v_1, v_2 are two different views. Given the camera calibration, each 2D point defines a line in 3D, $\mathbf{L}(i_{v_1})$ and $\mathbf{L}(j_{v_2})$. Now let $\mathbf{P}_{m_k}$ define the 3D point corresponding to the 3D node, which is the average between the closest points on the lines. The reconstruction cost is

$$C_{\text{rec}}(m_k) = \log\left(1 - \mathcal{F}\left(\text{dist}\left(\mathbf{L}(i_{v_1}), \mathbf{L}(j_{v_2})\right), \text{E}_{\text{3D}}\right)\right),$$

where E_{3D} is the maximum allowed 3D error. These edges are active, i.e., have a positive flow, when both originating 2D detections are also active. How to express this dependency in linear form is explained in Sect. 2.2.

Camera coherency edges (C_{coh}). They have a similar purpose to the reconstruction edges, but in this case their cost is related to the 3D distance between two 3D nodes from different camera pairs. We show a few of these edges in Fig. 2(b) in purple. Considering two camera pairs $k, l \in \mathcal{C}_v$, two 3D nodes m_k and n_l and their corresponding 3D points $\mathbf{P}_{m_k}$ and $\mathbf{P}_{n_l}$, we define the camera coherency edge cost as

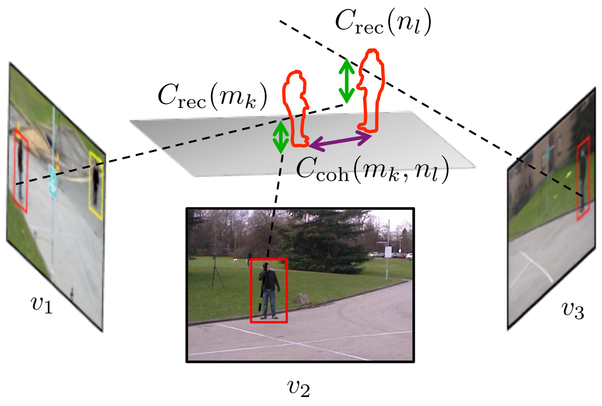
$$C_{\text{coh}}(m_k, n_l) = \log\left(1 - \mathcal{F}\left(\|\mathbf{P}_{m_k}, \mathbf{P}_{n_l}\|, \text{E}_{\text{3D}}\right)\right).$$

These edges are active when the two 3D nodes they connect are also active.

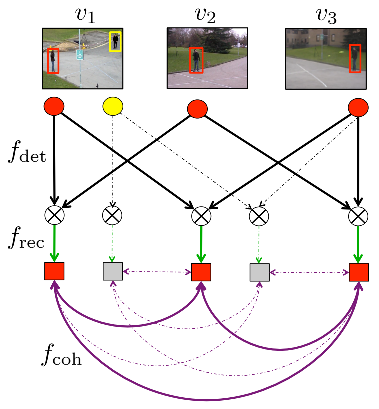
Temporal 3D edges ($C_{\text{t}_{\text{3D}}}$). The last type of edges are the ones that connect 3D nodes in several frames (shown in orange in Figure 2(b)). The connection is exactly the same as for the 2D nodes and their cost is defined as

$$C_{\text{t}_{\text{3D}}}(m_k, n_k) = \log\left(1 - \mathcal{F}\left(\frac{\|\mathbf{P}_{m_k} - \mathbf{P}_{n_k}\|}{\Delta t}, V^{\text{3D}}_{\text{max}}\right)\right),$$

where $V^{\text{3D}}_{\text{max}}$ is the maximum allowed speed in world coordinates. These edges are active when the two 3D nodes they connect are also active.
It is important to note that the 3D layer costs are always negative. To see this, recall that $\mathcal{F}()$ mapped a distance to a probability, therefore, the lower the distance it evaluates, the higher the probability will be and hence also the higher the negative cost. If the costs were positive, the solution would favor a separate trajectory for each camera and frame, because finding a common trajectory for all cameras and frames would activate these edges and therefore incur an extra cost to the solution. Instead, these edges act as *prizes* for the graph, so that having the same identity in 2 cameras is beneficial if the reconstruction, camera coherence and temporal 3D edges are sufficiently negative.



(a)



(b)

Figure 3: 3D layer edges: **(a)** The 2D nodes in each camera activate the reconstruction and camera coherency edges because they are assigned the same trajectory ID visualized in red. The reconstruction error C_{rec} is defined as the minimum line distance between projection rays. The camera coherency edges C_{coh} are defined as the 3D distance between putative reconstructions (illustrated as red silhouettes in 3D) from different camera pairs. **(b)** graph structure of the 3D layer: active edges are shown in continuous lines. The red 2D nodes (circles) activate the 3D nodes (square nodes) since they are assigned the same ID (product of flows equals one).

2.2. Linear programming

In the literature, multiple object tracking is commonly formulated as a Maximum A-Posteriori (MAP) problem. To convert it to a Linear Program (LP), its objective function is linearized with a set flow flags $f(i) = \{0, 1\}$ which indicate if an edge i is in the path of a trajectory or not [13, 25]. The proposed multi-layer graph can be expressed as a LP with the following objective function:

$$\begin{aligned}
\mathcal{T}* &= \underset{\mathcal{T}}{\operatorname{argmin}}\ \mathbf{C}^{\mathrm{T}}\mathbf{f} = \sum_i C(i)f(i) \\
&= \sum_{v=1}^{V}\sum_{i_v} C_{\text{in}}(i_v)f_{\text{in}}(i_v) + \sum_{v=1}^{V}\sum_{i_v} C_{\text{out}}(i_v)f_{\text{out}}(i_v) \\
&+ \sum_{v=1}^{V}\sum_{i_v} C_{\text{det}}(i_v)f_{\text{det}}(i_v) + \sum_{v=1}^{V}\sum_{i_v,j_v} C_{\text{t}}(i_v,j_v)f_{\text{t}}(i_v,j_v) \\
&+ \sum_{k\in\mathcal{C}_v}\sum_{m_k} C_{\text{rec}}(m_k)f_{\text{rec}}(m_k) \\
&+ \sum_{k\in\mathcal{C}_v}\sum_{l\in\mathcal{C}_v}\sum_{m_k,n_l} C_{\text{coh}}(m_k,n_l)f_{\text{coh}}(m_k,n_l) \\
&+ \sum_{k\in\mathcal{C}_v}\sum_{m_k,n_k} C_{\text{t}_{\text{3D}}}(m_k,n_k)f_{\text{t}_{\text{3D}}}(m_k,n_k) \qquad (1)
\end{aligned}$$

where $k, l \in \mathcal{C}_v$ are the indices of different camera pairs. The problem is subject to the following constraints:

- Edge capacities: we assume that each detection belongs only to one trajectory $f(i) = \{0, 1\}$. Since integer programming is NP-hard, we relax the conditions to obtain a linear program: $0 \leq f(i) \leq 1$. In the remainder of this paper all the conditions will be expressed in their relaxed form.
- Flow conservation at the 2D nodes: $f_{\text{in}}(i_v), f_{\text{out}}(i_v)$ indicate whether a trajectory starts or ends at node i_v.

$$\begin{aligned}
f_{\text{det}}(i_v) &= f_{\text{in}}(i_v) + \sum_{j_v} f_{\text{t}}(j_v, i_v) \\
f_{\text{det}}(i_v) &= \sum_{j_v} f_{\text{t}}(i_v, j_v) + f_{\text{out}}(i_v) \qquad (2)
\end{aligned}$$

- Activation for reconstruction edges: these 2D-3D connections have to be activated, i.e., have a positive flow, if their 2D originating nodes are also active. More formally, this imposes the following relationship:

$$f_{\text{rec}}(m_k) = f_{\text{det}}(i_{v_1})f_{\text{det}}(j_{v_2}) \qquad (3)$$

- Activation for the camera coherency edges: for 3D-3D connections we take a similar approach as for the reconstruction edges and define the flow to be dependent on the 3D nodes it connects:

$$f_{\text{coh}}(m_k, n_l) = f_{\text{rec}}(m_k)f_{\text{rec}}(n_l) \qquad (4)$$

- Activation for temporal 3D edges:

$$f_{\text{t}_{\text{3D}}}(m_k, n_k) = f_{\text{rec}}(m_k)f_{\text{rec}}(n_k) \qquad (5)$$

As we can see, the pairwise terms in Eqs. (3), (4) and (5) are non-linear. Let $f_{ab} = f_a f_b$ be a pairwise term consisting of two flows f_a and f_b . Using the fact that the flows are binary, we can encode the pairwise term with the following linear equations:

$$f_{ab} - f_a \leq 0 \qquad f_{ab} - f_b \leq 0 \qquad f_a + f_b - f_{ab} \leq 1.$$

Using this conversion, we can express the constraints in Eqs. (3), (4) and (5) in linear form. These constraints define the 3D layer of the graph as a *cascade of prizes*. Consider two 2D nodes on different cameras which belong to different trajectories. The question will be whether it is favorable to assign the same trajectory ID to both 2D nodes. The answer depends on the prize costs this assignment activates. When both 2D nodes are assigned the same trajectory ID, the corresponding 3D reconstruction edge is activated. If two 3D nodes from different camera pairs are activated, the camera coherency edge between them is activated, and the same will happen across time. This means that trajectories are assigned the same ID only if the reconstruction, camera coherency and temporal 3D costs are sufficiently negative to be beneficial to minimize the overall solution.

2.3. Multi-commodity flow formulation

The goal of the flow constraints defined in the previous section is to activate certain prize edges when two 2D nodes are activated *by the same object*. This means that in one graph we can only have a total flow of 1, which corresponds to one object. To that end, we create one more condition on the number of objects per camera:

$$0 \leq \sum_{i_v} f_{in}(i_v) \leq 1 \qquad 0 \leq \sum_{i_v} f_{out}(i_v) \leq 1 \quad \forall v \quad (6)$$

In order to deal with several objects, we use the multi-commodity flow formulation, well-known in traffic scheduling [18]. We create one graph for each object n to be tracked on the scene. Each graph has its own source and sink nodes, and each object is a commodity to be sent through the graph. The problem has now a much larger set of variables $\mathbf{f} = \left[\mathbf{f}^1 \dots \mathbf{f}^{N_{\text{obj}}}\right]$. Obviously, with no further restrictions, computing the global optimum would result in the same solution for all the instances of the graph, i.e., we would find the same trajectory for all the objects. Therefore, we need to create a set of *binding constraints* which prevent two trajectories from going through the same edges:

$$\sum_n f^n(i) \leq 1 \qquad n = 1 \dots N_{\text{obj}} \quad (7)$$

where $f^n(i)$ is the flow of object n going through the edge i. This set of binding constraints create a much complex linear program which cannot be solved with standard techniques. Nonetheless, the problem still has an interesting block-angular structure, which can be exploited. The problem consists of a set of small problems (or subproblems), one for each object, with the goal to minimize Eq. (1) subject to the constraints in Eq. (2)-(6). On the other hand, the set of complex binding constraints in Eq. (7) defines the master problem. This structure is fully exploited by the Dantzig-Wolfe decomposition method, which is explained in the next section, allowing the algorithm to find a solution with less computation time and with a better lower bound.

3. Branch-and-price for multi-commodity flow

Branch-and-price is a combinatorial optimization method for solving large scale integer linear problems. It is a hybrid method of column generation and branching.

Column generation: Dantzig-Wolfe decomposition. The principle of decomposition is to divide the constraints of an integer problem into a set of "easy constraints" and a set of "hard constraints". The idea is that removing the hard constraints results in several subproblems which can be easily solved by k-shortest paths, simplex, etc. Let us rewrite our original minimum cost flow problem:

$$\min_{\mathbf{f}} \mathbf{C}^{\mathrm{T}}\mathbf{f} = \sum_{n=1}^{N_{\text{obj}}} (\mathbf{c}^n)^{\mathrm{T}}\mathbf{f}^n$$

subject to:

$$\mathbf{A}_1\mathbf{f} \preceq \mathbf{b}_1 \qquad \mathbf{A}_2^n\mathbf{f}^n \preceq \mathbf{b}_2^n \qquad 0 \preceq \mathbf{f} \preceq 1$$

where $(\mathbf{A}_1, \mathbf{b}_1)$ represent the set of hard constraints Eq. (7), and $(\mathbf{A}_2, \mathbf{b}_2)$ the set of easy constraints, Eqs. (3)-(6), which are defined independently for each object $n = 1 \dots N_{\text{obj}}$. The idea behind Dantzig-Wolfe decomposition is that the set $\mathcal{T}^* = \{f \in \mathcal{T} : f \text{ integer}\}$, with $\mathcal{T}$ bounded, is represented by a finite set of points, i.e., a bounded convex polyhedron is represented as a convex combination of its extreme points. The *master problem* is then defined as:

$$\min_{\lambda} \sum_{n=1}^{N_{\text{obj}}} (\mathbf{c}^n)^{\mathrm{T}} \sum_{j=1}^{J} \mathbf{x}_j^n \lambda_j^n$$

subject to:

$$\sum_n \mathbf{A}_1^n \sum_{j=1}^{J} \mathbf{x}_j^n \lambda_j^n \preceq \mathbf{b}_1 \qquad \sum_{j=1}^{J} \lambda_j^n = 1 \qquad 0 \leq \lambda_j^n \leq 1$$

where $\mathbf{f}^n = \sum_{j=1}^{J} \lambda_j^n \mathbf{x}_j^n$ and $\{\mathbf{x}_j\}_{j=1}^{J}$ are the extreme points of a polyhedra. This problem is solved using column generation (Algorithm 1). The advantage of this formulation is that the N_{obj} column generation subproblems can be solved independently and therefore in parallel. We use the parallel implementation found in [17] which is based on [18]. Furthermore, decomposition methods strengthen the bound of the relaxed LP problem.

Branching. Typically in multi-commodity flow problems, the solution is not guaranteed to be integer. Nonetheless, once we find the fractional solution, we can use branching schemes to find an integer solution. This mixture of column generation and branching is called *branch-and-price*. For more details we refer to [1, 2].

4. Experimental results

In this section we show the tracking results of the proposed method on two key problems in computer vision,

Algorithm 1 Column generation

```
while Restricted master problem new columns $> 0$ do
    1. Select a subset of columns corresponding to $\lambda_j^n$ which form what is called the *restricted master problem*
    2. Solve the restricted problem with the chosen method (e.g. simplex).
    3. Calculate the optimal dual solution $\mu$
    4. Price the rest of the columns with $\mu(\mathbf{A}_1^n\mathbf{f}^n - \mathbf{b}_1^n)$
    5. Find the columns with negative cost and add them to the restricted master problem. This is done by solving $N_{\text{obj}}$ column generation *subproblems*.
        $\min_{\mathbf{f}} \quad (\mathbf{c}^n)^{\mathrm{T}}\mathbf{f}^n + \mu(\mathbf{A}_1^n\mathbf{f}^n - \mathbf{b}_1^n) \quad \text{s.t.} \quad \mathbf{A}_2^n\mathbf{f}^n \preceq \mathbf{b}_2^n$
end while
```

namely multi-camera multiple people tracking and 3D human pose tracking. We compare our method with the following approaches for multi-view multiple object tracking:

- Greedy Tracking-Reconstruction (GTR): first tracking is performed in 2D in a frame-by-frame basis using bipartite graph matching, and then 3D trajectories are reconstructed from the information of all cameras.
- Greedy Reconstruction-Tracking (GRT): first the 3D positions are reconstructed from all cameras. In a second step, 3D tracking is performed in a frame-by-frame basis using bipartite graph matching.
- Tracking-Reconstruction (TR): first tracking is performed in 2D using [25] and then 3D trajectories are recovered as in GTR.
- Reconstruction-Tracking (RT): first the 3D positions are reconstructed as in GRT and then 3D tracking is performed using [25].

Tests are performed on two publicly available datasets [7, 20] and comparison with existing state-of-the-art tracking approaches is done using the CLEAR metrics [9], DA (detection accuracy), TA (tracking accuracy), DP (detection precision) and TP (tracking precision).

4.1. Multi-camera multiple people tracking

In this section we show the tracking results of our method on the publicly available PETS2009 dataset [7] dataset, a scene with several interacting targets. Detections are obtained using the Mixture of Gaussians (MOG) background subtraction. For all experiments, we set $B_f = 0.3$, $\mathrm{E}_{3\mathrm{D}} = 0.5$ m. which amounts for the diameter of a person, $V_{\text{max}}^{\text{2D}} = 250$ pix/s. and $V_{\text{max}}^{\text{3D}} = 6$ m/s. which is the maximum allowed speed for the pedestrians. Note, that for this particular dataset, we can infer the 3D position of a pedestrian with only one image since we can assume $z = 0$. Since we evaluate on view 1, and the second view we use does

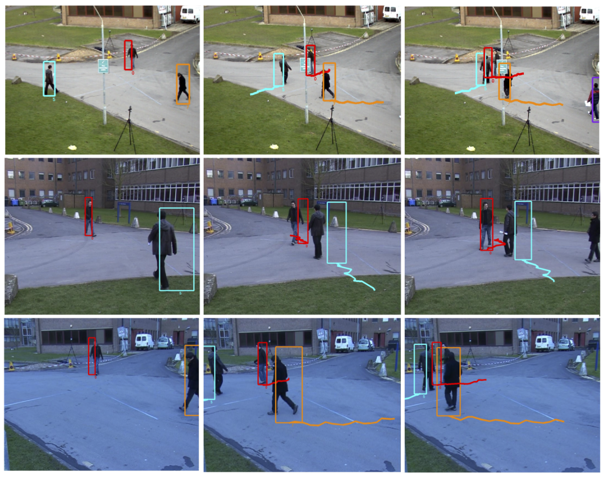


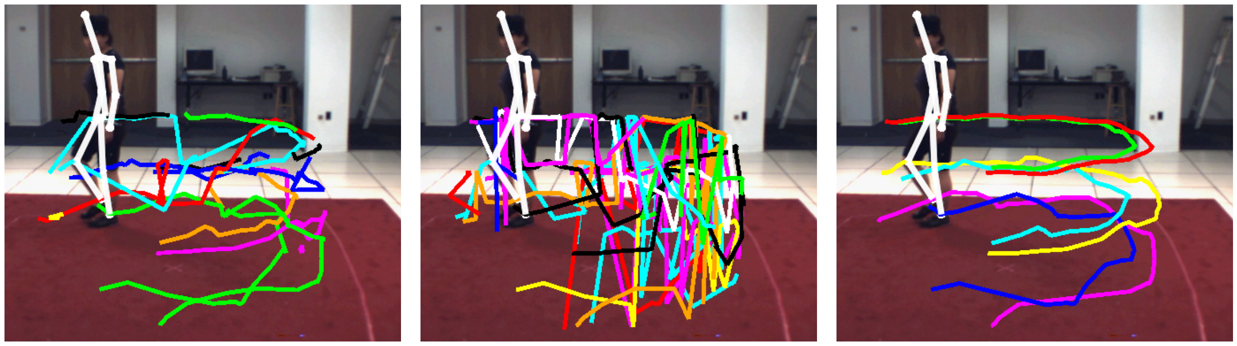
Figure 4: Results on the PETS sequence, tracking with 3 camera views. Although there are clear 2D-3D inaccuracies the proposed method is able to track the red pedestrian which is occluded in 2 cameras during 22 frames.

not show all the pedestrians, it would be unfair for the RT and GRT methods to only reconstruct pedestrians visible in both cameras. Therefore, we consider the detections of view 1 as the main detections and only use the other cameras to further improve the 3D position. We also compare our results to monocular tracking using [25] and multi-camera tracking with Probability Occupancy Maps and Linear Programming [4]. As we can see in the results with 2 cam-

	DA	TA	DP	TP	miss
Zhang et al. [25] (1)	68.9	65.8	60.6	60.0	28.1
GTR(2)	51.9	49.4	56.1	54.4	31.6
GRT (2)	64.6	57.9	57.8	56.8	26.8
TR (2)	66.7	62.7	59.5	57.9	24.0
RT (2)	69.7	65.7	61.2	60.2	25.1
Berclaz et al. [4] (5)	76	75	62	**62**	–
Proposed (2)	**78.0**	**76**	**62.6**	60	**16.5**
TR (3)	48.5	46.5	51.1	50.3	20
RT (3)	56.6	51.3	54.5	52.8	23.5
Proposed (3)	73.1	71.4	55.0	53.4	**12.9**

Table 1: PETS2009 L1 sequence. Comparison of several methods tracking on a variable number of cameras (indicated in parenthesis).

era views, Table 1, the proposed algorithm outperforms all other methods. In general, TR and RT methods perform better than their counterparts GRT and GTR, since matching across time with Linear Programming is robust to short occlusions and false alarms. Nonetheless, it still suffers from long term occlusions. In contrast, our method is more powerful than existing approaches when dealing with missing and noisy data, with miss detection rates 8.5% to 15% lower than other methods. Notably, our method also out-



(a) Tracking-Reconstruction (b) Reconstruction-Tracking (c) Proposed method

Figure 5: Even with 40% of outliers our method 5(c) can recover the trajectories almost error free during all the sequence. This is in contrast to 5(a) and 5(b) that struggle with the ambiguities generated by the outliers.

performs [4] in accuracy, even though our results are only with 2 cameras instead of 5. When using 3 cameras, the 2D-3D inaccuracies become more apparent since the detections of the third camera project badly on the other two cameras (see Fig. 4). Interestingly, RT and TR methods are greatly affected by these inaccuracies, while the proposed method is more robust and still able to further reduce the missed detections by 4.6%. In Fig. 4, we show an example where a pedestrian (red) is occluded in two of the three views for a length of 22 frames. The RT method is unable to recover any 3D position, and therefore loses track of the pedestrian. The TR method tries to track the pedestrian in one view, but the gap is too long and fails to finally recover the whole 3D trajectory. The proposed method overcomes the long occlusion and the noisy 2D-3D correspondences to recover the full trajectory. We obtain a 13.5% better accuracy than RT(3) which further proves the advantages of our approach.

4.2. Human Motion

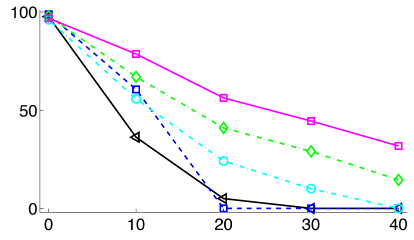
We also tested our algorithm for the problem of human pose tracking on the publicly available human motion database *HumanEva* [20]. The problem we consider here is the following: given a set of 2D joint locations in two cameras, the goal is to link the locations across time and across cameras at every frame to reconstruct the sequence of poses. In these experiments, we use only two cameras at a reduced frame rate of 10 fps to reconstruct the 3D poses. To obtain joint locations in the image we project the ground truth 3D data using the known camera parameters. The parameters used are: $B_f = 0.3$, $E_{3D} = 0.01$ mm., $V_{\max}^{2D} = 400$ pix/s. and $V_{\max}^{3D} = 3$ m/s. We study the robustness of our algorithm to missing data and outliers. Missing data often occurs due to occlusions while outliers appear as the result of false detections.

Missing data: To simulate missing data we increasingly removed percentages of the 2D locations from 0 to 40%. As it can be seen in Fig. 6(a) our proposed method outperforms all other baselines and brings significant improvement. In Fig. 7 we show the trajectories of the lower body reconstructed with our method with a 20% of missing data. The 3D error for our method stays below 5mm. whereas it goes up to 10mm. for the other methods.

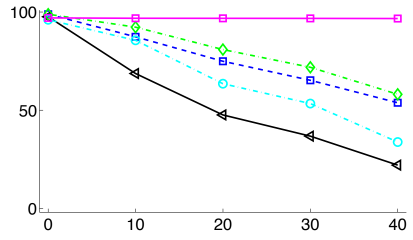
Outliers: We added from 0% to 40% of uniformly distributed outliers in windows of 15×15 pixels centered at randomly selected 2D joint locations. Again, our method shows a far superior performance as the percentage of outliers increases, see Fig. 6(b). Notably, our method performs equally well independently of the number of outliers. Since outliers are uncorrelated across cameras they produce lower prizes in the 3D layer of our graph and are therefore correctly disregarded during optimization. This clearly shows the advantage of globally exploiting temporal and 3D coherency information together. Here, the 3D error is only 2mm. for our method. Furthermore, in Fig. 6(c), we show the identity switches for increasing number of outliers. Our method is the only one that is virtually unaffected by the outliers, an effect that is also shown in Fig. 5. This last result is particularly important for pose tracking as ID switches result in totally erroneous pose reconstructions.

5. Conclusions

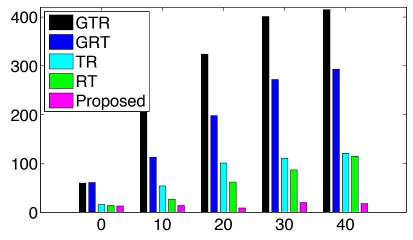
In this paper, we presented a new algorithm to jointly track multiple target in multiple views. The novel graph structure captures both temporal correlations between objects as well as spatial correlations enforced by the configuration of the cameras, and allows us to solve the problem as one global optimization. To find the global optimum, we used the powerful tool of branch-and-price, which allows us to exploit the special block-angular structure of the program to reduce computational time as well as to find a better lower bound. We tested the performance of the proposed approach on two key problems in computer vision: multiple people tracking and 3D human pose tracking. We outperform state-of-the-art approaches which proves the strength



(a) Tracking accuracy (%) with missing data

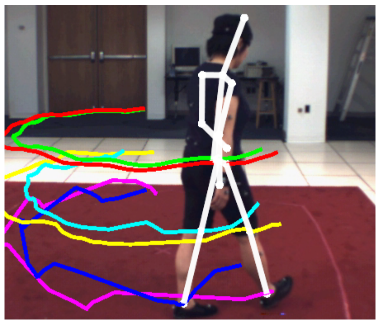


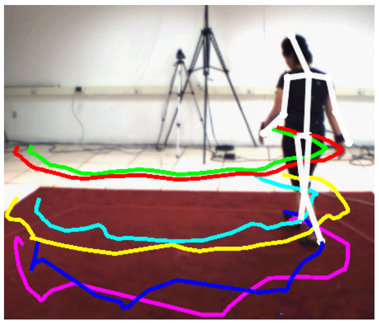
(b) Tracking accuracy (%) with outliers



(c) ID switches with outliers

Figure 6: Robustness evaluation: simulation of increasing missing data 6(a) and increasing outliers 6(b),6(c).


(a) Camera 1


(b) Camera 2

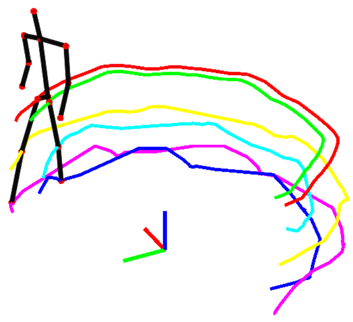

(c) 3D trajectories

Figure 7: Proposed method with 20% of missing data. Note that the trajectories are assigned the same ID in both views.

of combining 2D and 3D constraints in a single global optimization. The proposed formulation can be of considerable interest to model complex dependencies which arise in a wide range of computer vision problems.

References

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network flows: Theory, algorithms and applications*. Prentice Hall, 1993. 5

[2] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46, 1996. 2, 5

[3] J. Berclaz, F. Fleuret, and P. Fua. Robust people tracking with global trajectory optimization. *CVPR*, 2006. 1

[4] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *TPAMI*, 2011. 2, 6, 7

[5] D. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999. 2

[6] P. Chardaire and A. Sutter. A decomposition method for quadratic zero-one programming. *Management Science*, 1995. 2

[7] J. Ferryman. Pets 2009 dataset: Performance and evaluation of tracking and surveillance. 2009. 6

[8] H. Jiang, S. Fels, and J. Little. A linear programming approach for multiple object tracking. *CVPR*, 2007. 2

[9] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, M. Boonstra, V. Korzhova, and J. Zhang. Framework for performance evaluation for face, text and vehicle detection and tracking in video: data, metrics, and protocol. *TPAMI*, 31(2), 2009. 6

[10] Z. Khan, T. Balch, and F. Dellaert. Mcmc-based particle filtering for tracking a variable number of interacting targets. *TPAMI*, 2005. 1

[11] N. Komodakis, N. Paragios, and G. Tziritas. Mrf optimization via dual decomposition: Message-passing revisited. *ICCV*, 2007. 2

[12] L. Leal-Taixé, M. Heydt, A. Rosenhahn, and B. Rosenhahn. Automatic tracking of swimming microorganisms in 4d digital in-line holography data. *IEEE Workshop on Motion and Video Computing (WMVC)*, 2009. 1

[13] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. *ICCV. 1st Workshop on Modeling, Simulation and Visual Analysis of Large Crowds*, 2011. 2, 3, 4

[14] S. Pellegrini, A. Ess, and L. van Gool. Improving data association by joint modeling of pedestrian trajectories and groupings. *ECCV*, 2010. 2

[15] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. *CVPR*, 2011. 2

[16] A. Poore. Multidimensional assignment formulation of data association problems rising from multitarget and multisensor tracking. *Computational Optimization and Applications*, 1994. 2

[17] J. Rios. http://sourceforge.net/apps/wordpress/dwsolver/. 5

[18] J. Rios and K. Ross. Massively parallel dantzig-wolfe decomposition applied to traffic flow scheduling. *Journal of Aerospace Computing, Information, and Communication*, 7(1), 2010. 2, 5

[19] H. Shitrit, J. Berclaz, F. Fleuret, and P. Fua. Tracking multiple people under global appearance constraints. *ICCV*, 2011. 2

[20] L. Sigal, A. Balan, and M. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *IJCV*, 87(1):4–27, 2010. 6, 7

[21] L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching:models and global optimization. *ECCV*, 2008. 2

[22] Z. Wu, N. Hristov, T. Kunz, and M. Betke. Tracking-reconstruction or reconstruction-tracking? comparison of two multiple hypothesis tracking approaches to interpret 3d object motion from several camera views. *WACV*, 2009. 2

[23] Z. Wu, T. Kunz, and M. Betke. Efficient track linking methods for track graphs using network-flow and set-cover techniques. *CVPR*, 2011. 2

[24] Q. Yu, G. Medioni, and I. Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. *CVPR*, 2007. 2

[25] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. *CVPR*, 2008. 2, 4, 6