On Pairwise Costs for Network Flow Multi-Object Tracking

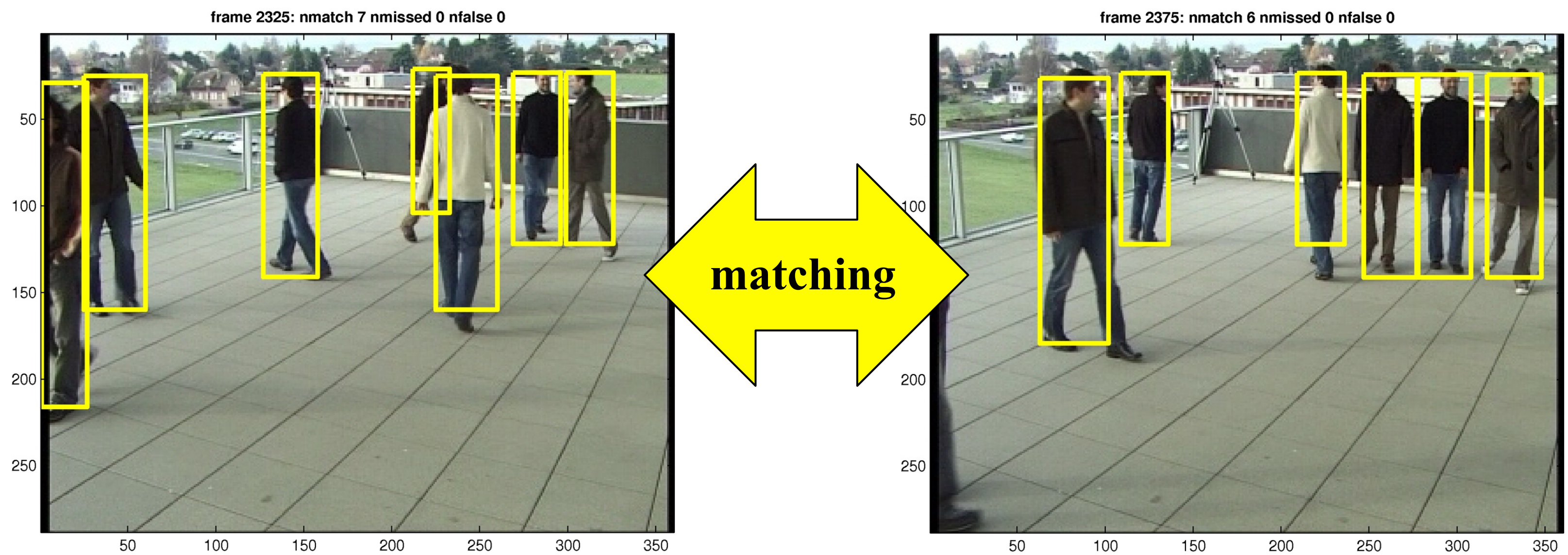
Visesh Chari, Simon Lacoste-Julien, Ivan Laptev, and Josef Sivic

CVPR 2015

Detect Then Track

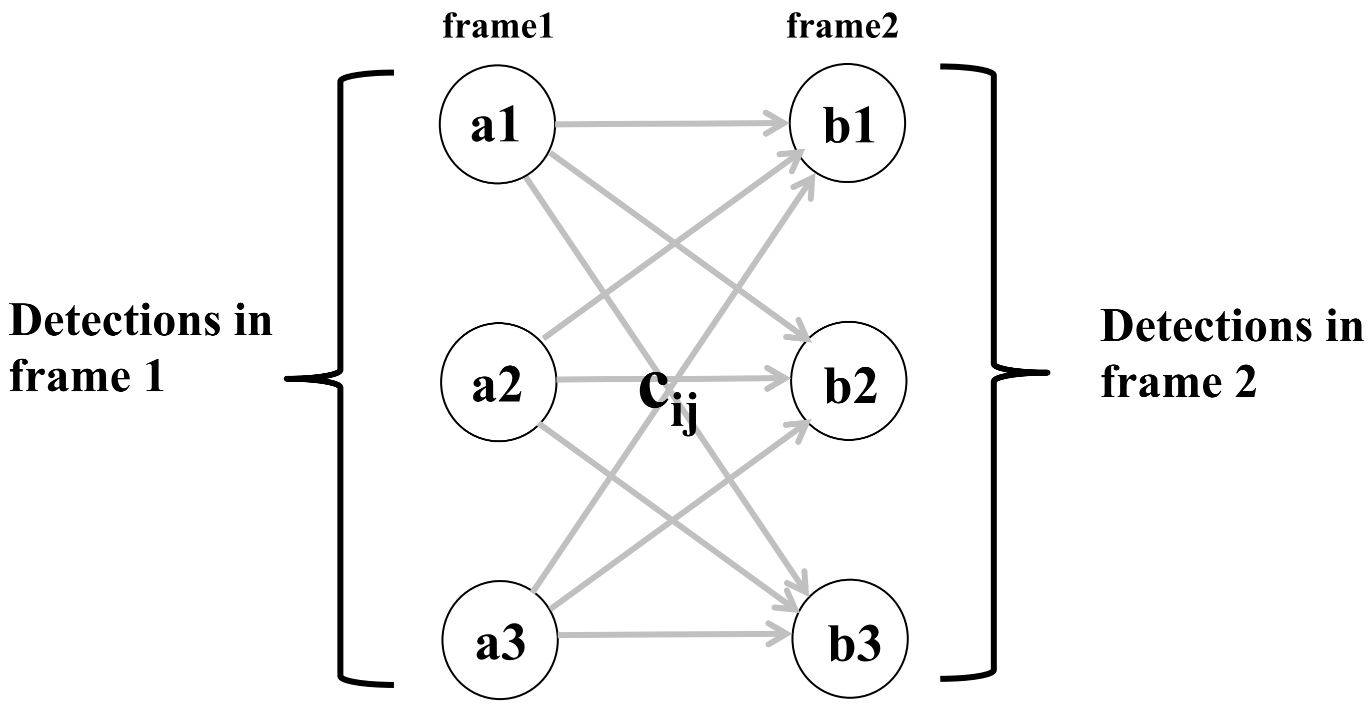
Consider a two-stage approach to multi-object tracking

- Detect objects in each frame of a sequence
- Determine interframe correspondences between them to label objects and discover their trajectories



Robert Collins
Penn State

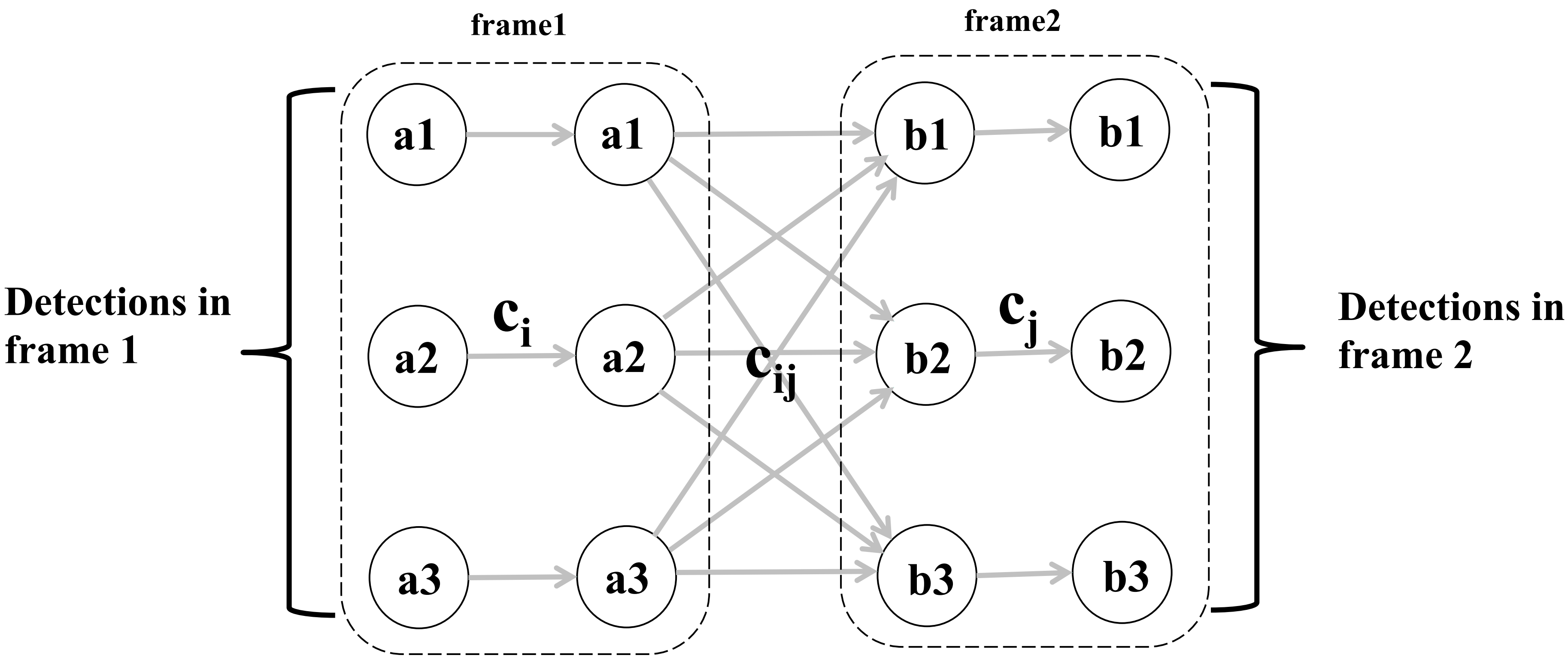
Data Association



c_{ij} = cost to associate detection i with detection j (lower cost is better)

Robert Collins
Penn State

Data Association

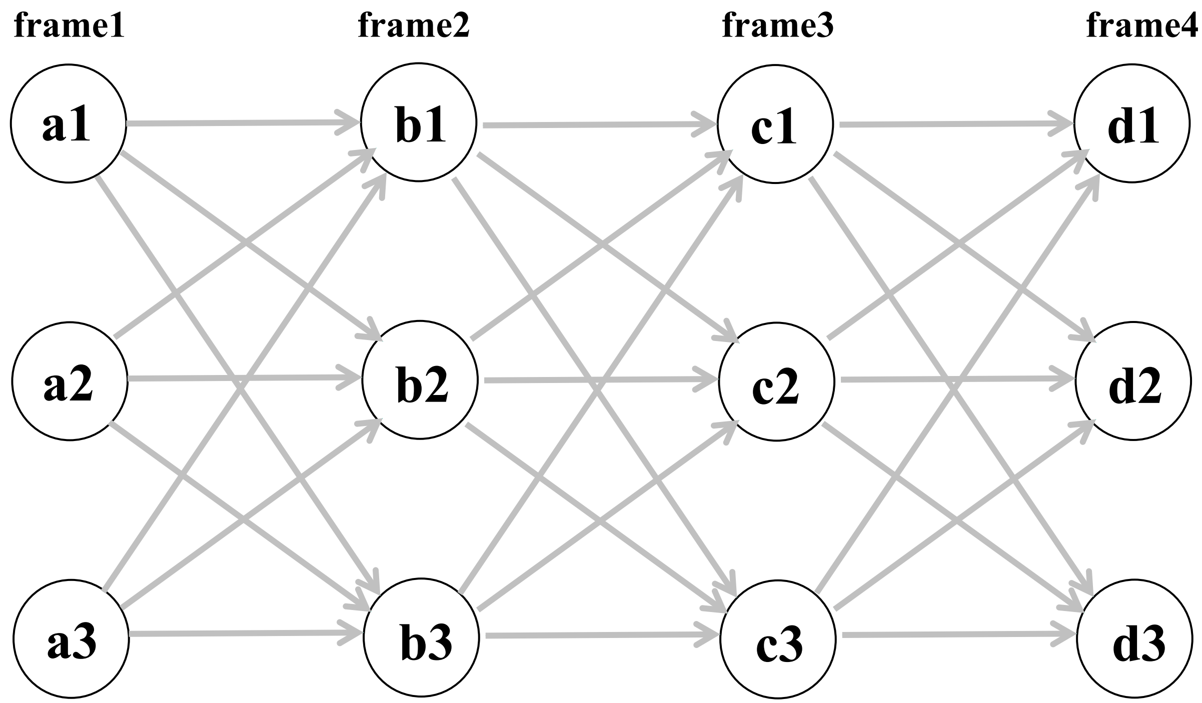


c_{ij} = cost to associate detection i with detection j

c_i = cost to include detection i in the solution (e.g. negative detector confidence)

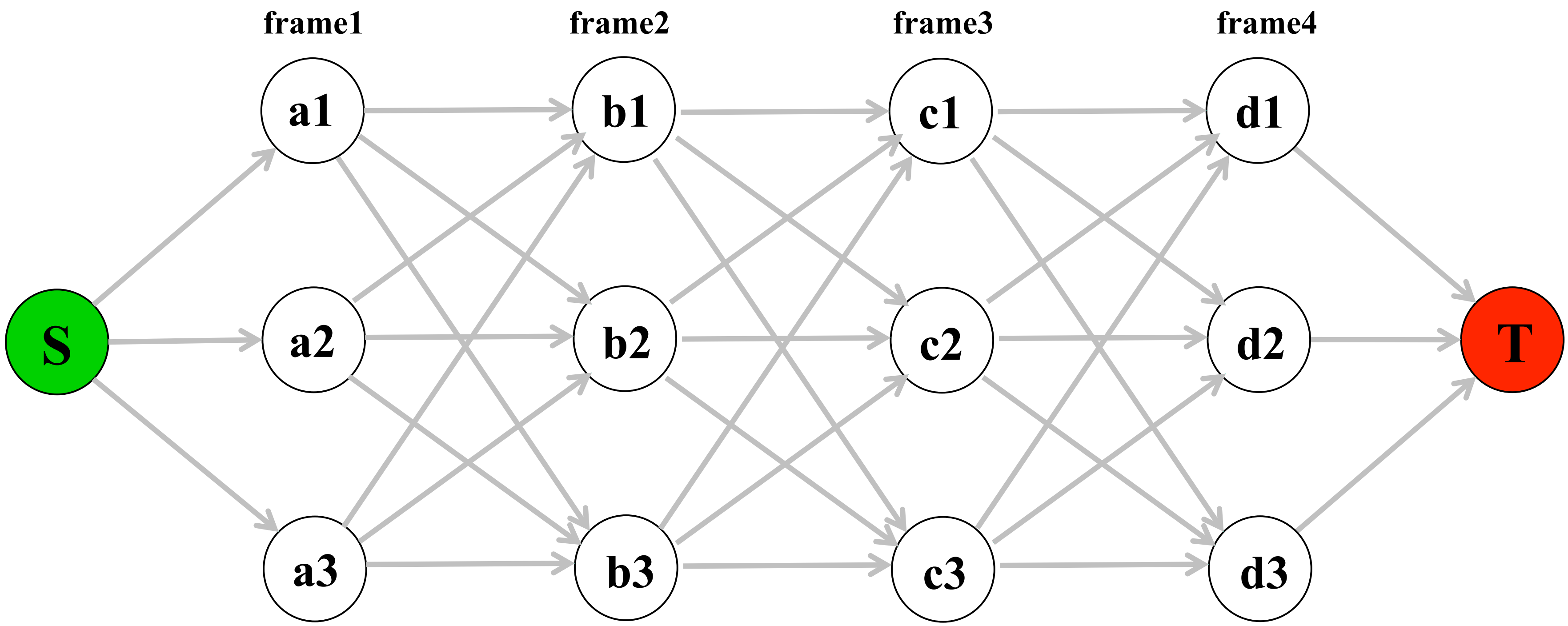
Robert Collins
Penn State

Multi-frame Data Association



Robert Collins
Penn State

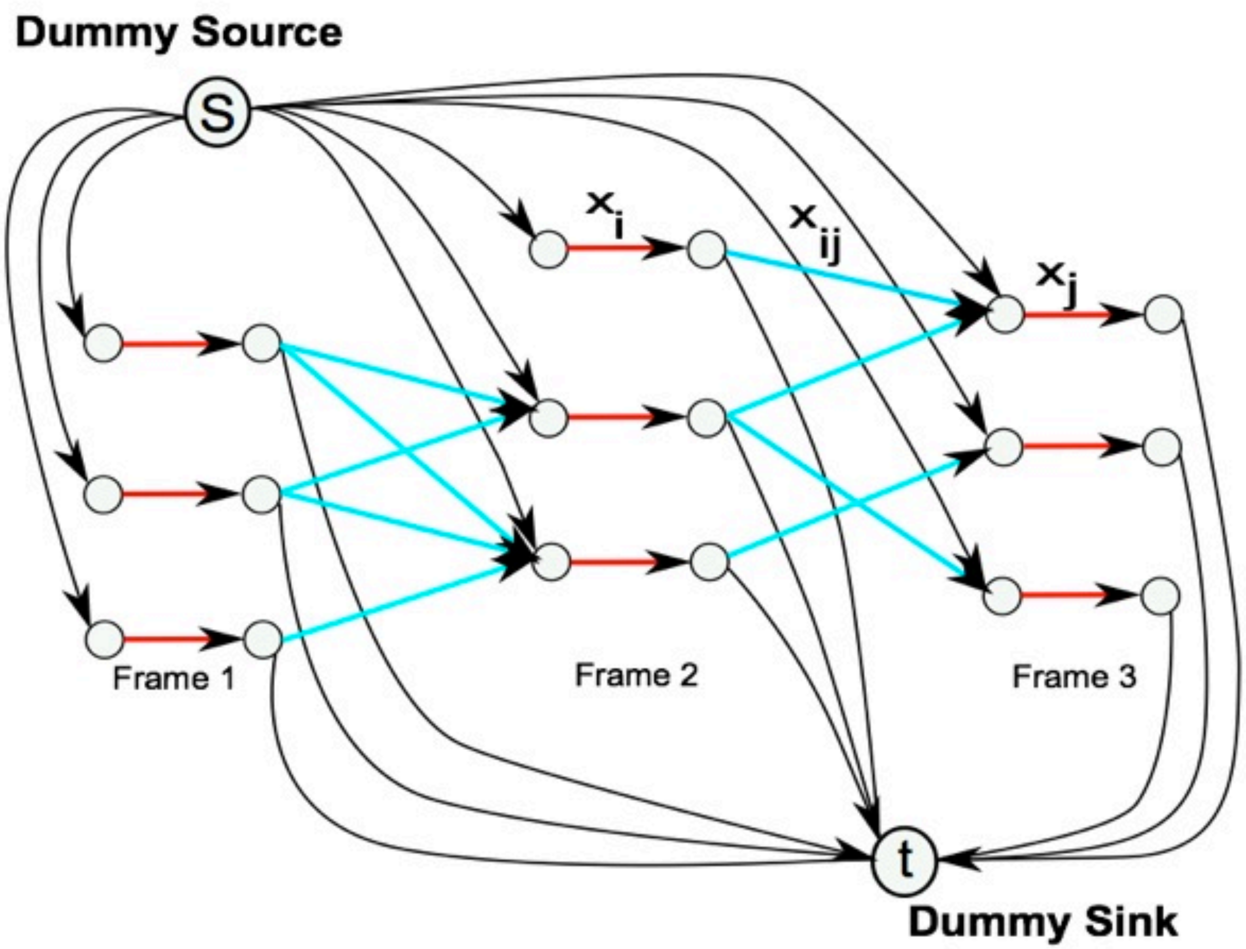
Network Flow Approach



Pump K units of flow from S to T while minimizing cost.
Note: in practice, S and T are connected to all detections, since a trajectory can start or end in any frame.

Robert Collins
Penn State

Network Flow Approach



Involves solving for binary variables *x* on each edge.
Variable is 1 if edge is part of the solution, 0 if not.

Robert Collins
Penn State

Formulated as ILP

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & \textstyle\sum_i c_i x_i + \sum_{ij \in E} c_{ij} x_{ij} \\
\text{s.t.} \quad & \left.\begin{aligned}
& 0 \le x_i \le 1\,,\, 0 \le x_{ij} \le 1 \\
& \sum_{i\,:\,ij \in E} x_{ij} = x_j = \sum_{i\,:\,ji \in E} x_{ji} \\
& \sum_i x_{it} = K = \sum_i x_{si}
\end{aligned}\right\} \mathbf{x} \in \mathrm{FLOW}_K \\
& x_i\,,\, x_{ij} \quad \text{are} \quad \text{integer.}
\end{aligned}
$$

Formulated as ILP

$$\min_{\mathbf{x}} \quad \sum_i c_i x_i + \sum_{ij \in E} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \left.\begin{array}{c} 0 \le x_i \le 1 \,,\, 0 \le x_{ij} \le 1 \\ \boxed{\sum_{i:ij \in E} x_{ij} = x_j = \sum_{i:ji \in E} x_{ji}} \\ \boxed{\sum_i x_{it} = K = \sum_i x_{si}} \end{array}\right\} \mathbf{x} \in \mathrm{FLOW}_K$$

$$x_i \,,\, x_{ij} \quad \text{are} \quad \text{integer.}$$

Enforce path continuity

K units of flow

Robert Collins
Penn State

Formulated as ILP

$$\min_{\mathbf{x}} \quad \sum_i c_i x_i + \sum_{ij \in E} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \left.\begin{array}{l} \boxed{0 \le x_i \le 1\,,\, 0 \le x_{ij} \le 1} \\ \displaystyle\sum_{i\,:\,ij \in E} x_{ij} = x_j = \sum_{i\,:\,ji \in E} x_{ji} \\ \displaystyle\sum_i x_{it} = K = \sum_i x_{si} \end{array}\right\} \mathbf{x} \in \mathrm{FLOW}_K$$

$$\boxed{x_i\,,\, x_{ij} \quad \text{are} \quad \text{integer.}}$$

Implies each x must be 0 or 1

Formulated as ILP

$$\min_{\mathbf{x}} \quad \textstyle\sum_i c_i x_i + \sum_{ij \in E} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \left.\begin{array}{l} \boxed{0 \le x_i \le 1\,, 0 \le x_{ij} \le 1} \\ \displaystyle\sum_{i:ij \in E} x_{ij} = x_j = \sum_{i:ji \in E} x_{ji} \\ \displaystyle\sum_i x_{it} = K = \sum_i x_{si} \end{array}\right\} \mathbf{x} \in \mathrm{FLOW}_K$$

$$\boxed{x_i\,, x_{ij} \quad \text{are} \quad \text{integer.}}$$

Implies each x must be 0 or 1

Fun fact: we can drop (relax) this constraint to get an LP rather than ILP, and still be guaranteed an integer 0,1 solution.
(due to totally unimodular constraints)

Network Flow Approach

Pros:

- Efficient solution (guaranteed polynomial time algorithms)
- Uses all frames to achieve a global batch solution

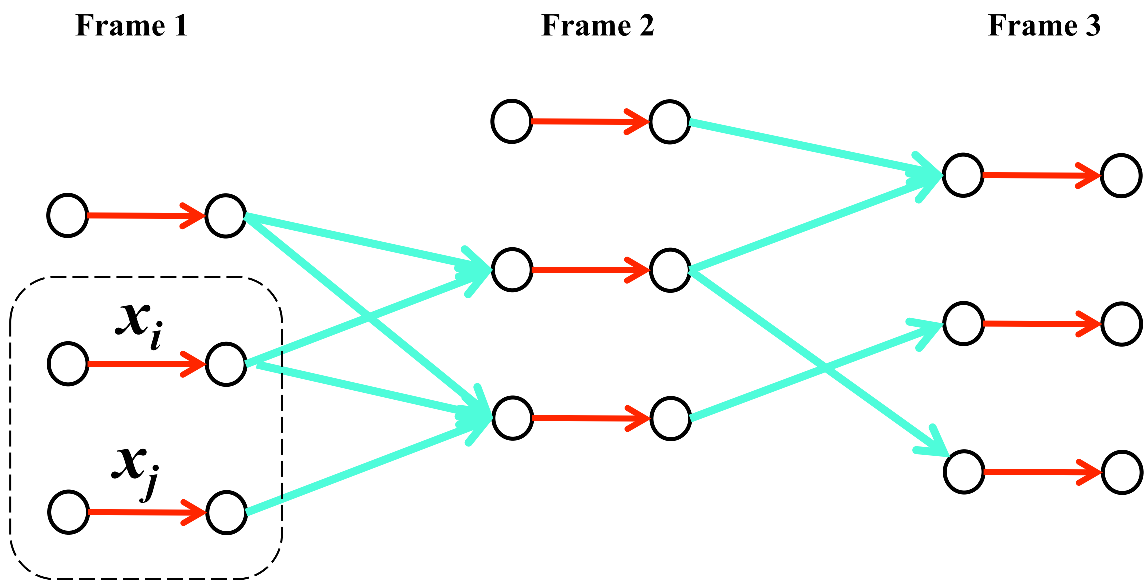
Cons:

- Cost function is limited to unary and pairwise costs defined over nodes
- Cannot represent higher-order terms, such as pairwise costs that are defined over (pairs of) edges

This paper: add ability to use pairwise edge costs

Robert Collins
Penn State

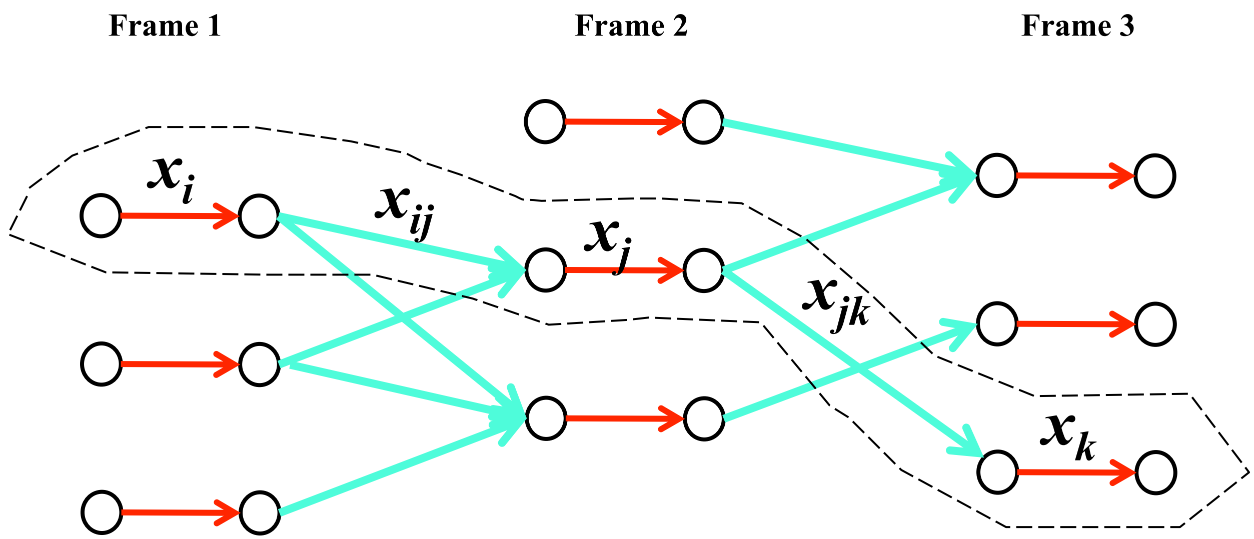
Examples: Pairwise Costs on Edges



$$\mathbf{Cost}(x_i\,, x_j) = q_{ij}\, x_i\, x_j$$

Motivation: Discourage two overlapping detections from both being part of the solution.
Aka: **non-maximum suppression**.

Examples: Pairwise Costs on Edges

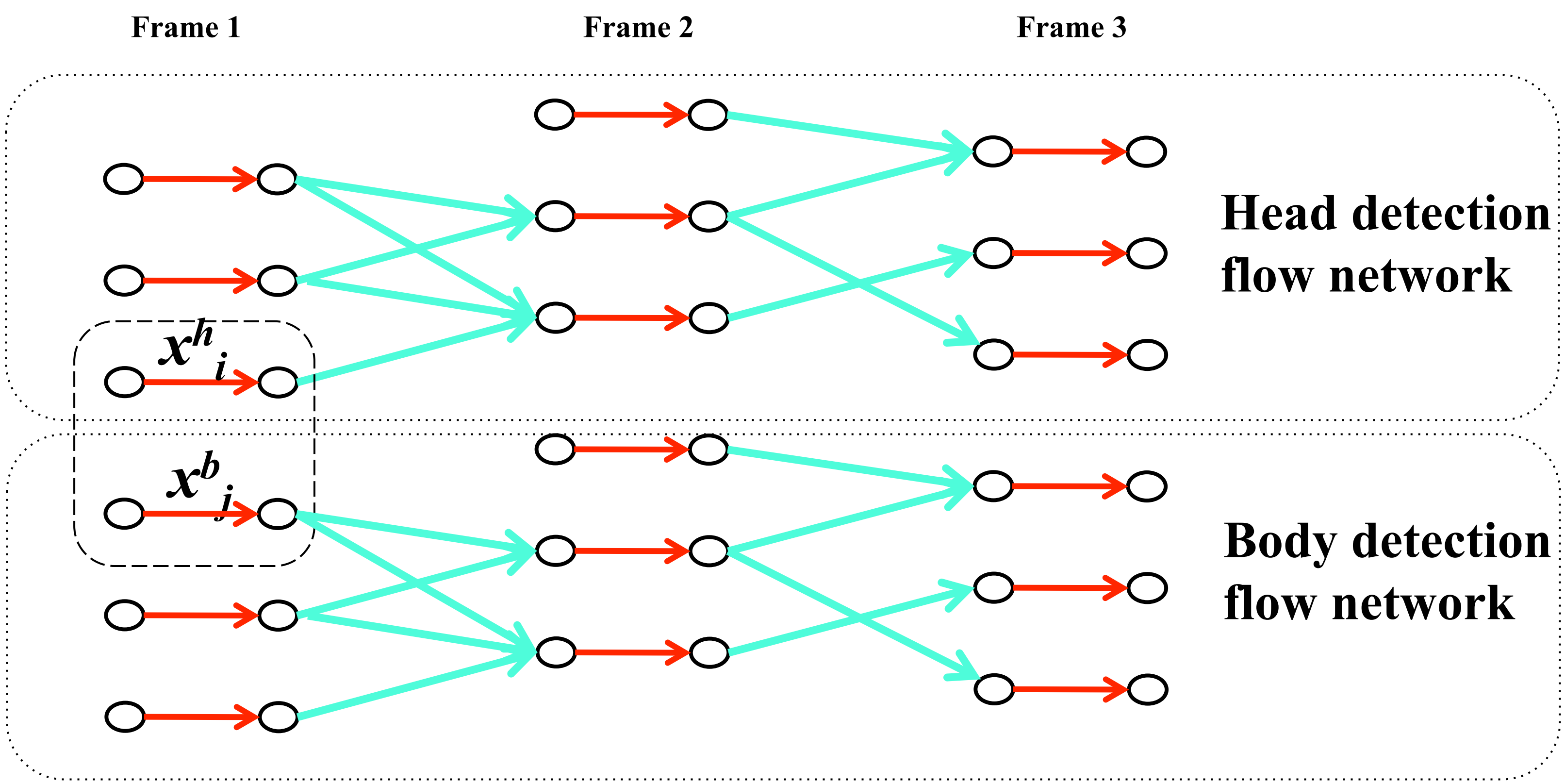


Cost(x_{ij} , x_{jk}) = $q_{ijk}\, x_{ij}\, x_{jk}$

Motivation: Encourage smooth trajectories, e.g. constant velocity motion.
This is not considered in this paper.

Robert Collins
Penn State

Examples: Pairwise Costs on Edges



$$\mathrm{Cost}(x^h_i, x^b_j) = -q_{ij}\, x^h_i\, x^b_j$$

Motivation: Encourage two compatible detections of different types to both be part of the solution.
E.g. co-occuring head and body detections

Robert Collins
Penn State

Approach: Form an IQP

relabel all vars as $z_1,\ldots,z_M$

Let $\mathbf{z} = [z_1, \ldots, z_M]$

$$= [\underbrace{x_1, \ldots, x_i, \ldots, x_n}_{\text{n detection vars}}, \underbrace{x_{11}, \ldots, x_{ij}, \ldots, x_{mn}}_{\text{M-n connection vars}}]$$

Form the integer quadratic program:

$$\begin{aligned} \min_{\mathbf{z}} \quad & \mathbf{c}^\top \mathbf{z} + \mathbf{z}^\top \mathbf{Q}\mathbf{z} \\ \text{s.t.} \quad & \mathbf{z} \in \mathrm{FLOW}_K \\ & \mathbf{z} \quad \text{integer} \end{aligned}$$

c is linear costs (on edges)
Q is sparse matrix of quadratic costs (on edge pairs)

PROBLEM!

Integer Quadratic Programs are **NP-hard** in general

This paper therefore discusses approximate solution methods.

Robert Collins
Penn State

Solution Approach

$$\min_{\mathbf{z}} \quad \mathbf{c}^\top \mathbf{z} + \mathbf{z}^\top \mathbf{Q} \mathbf{z}$$
$$\text{s.t.} \quad \mathbf{z} \in \text{FLOW}_K$$
$$\cancel{\mathbf{z} \quad \text{integer}}$$

Relax the integer constraints
Unfortunately, sol'n vars can take values between 0 and 1.
"Fix up" the solution so that all variables are 0 or 1, while maintaining feasibility (satisfy the Flow constraints).

Robert Collins
Penn State

Solution Method 1

Modify diagonal terms of Q so that it is positive semidefinite

$$Q_{ii}^{\text{new}} = \sum_{j \neq i} |Q_{ij}^{\text{old}}|$$

and adjust linear costs c_i to keep same objective function value

$$c_i^{\text{new}} = c_i - Q_{ii}^{\text{new}} + Q_{ii}^{\text{old}}$$

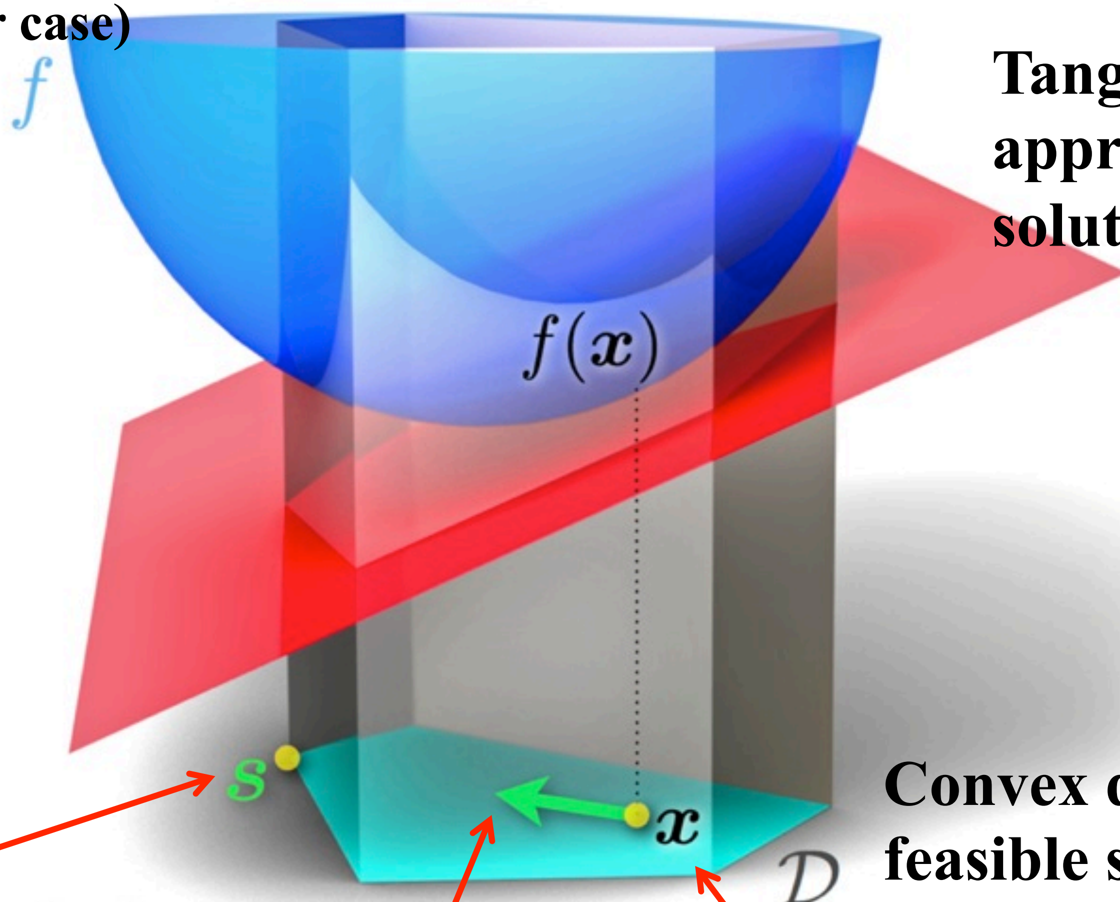
Note: $z_i^2 = z_i$

Problem is now convex, and a global solution can be found efficiently, e.g. by gradient descent or by the Frank-Wolfe algorithm which iteratively minimizes a linearization of the convex quadratic problem.

Frank-Wolfe Algorithm

Convex function
(quadratic in our case)

Tangent plane approx at current solution estimate



Convex domain of feasible solutions.

argmin $\mathbf{s}^T \nabla f(\mathbf{x}_k)$
$\mathbf{s} \in \mathcal{D}$

$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \gamma(\mathbf{s}_k - \mathbf{x}_k)$

current estimate x_k

Robert Collins
Penn State

Frank-Wolfe Algorithm

$$\begin{array}{ll} \min\limits_{\mathbf{z}} & \mathbf{c}^\top \mathbf{z} + \mathbf{z}^\top \mathbf{Q} \mathbf{z} \\ \text{s.t.} & \mathbf{z} \in \mathrm{FLOW}_K \end{array}$$

Let current solution estimate be z^*

Tangent is $(\mathbf{c} + (\mathbf{Q} + \mathbf{Q}^\top)\mathbf{z}^*)^\top \mathbf{z}$

To find s (= argmin z) we solve the network flow problem

$$\begin{array}{ll} \min\limits_{\mathbf{z}} & (\mathbf{c} + (\mathbf{Q} + \mathbf{Q}^\top)\mathbf{z}^*)^\top \mathbf{z} \\ \text{s.t.} & \mathbf{z} \in \mathrm{FLOW}_K \,. \end{array}$$

Solution Method 2

Introduce additional variables $u_{ij} = z_i z_j$ and constraints in order to form an equivalent integer linear program.

$$
\begin{aligned}
\min_{\mathbf{z},\mathbf{u}} \quad & \mathbf{c}^\top \mathbf{z} + \mathbf{q}^\top \mathbf{u} \\
\text{s.t.} \quad & \mathbf{z} \in \mathrm{FLOW}_K \\
& \left.\begin{aligned} 0 \le u_{ij} \le 1,\ \forall ij \in \mathcal{Q} \\ u_{ij} \le z_i\,,\ u_{ij} \le z_j \\ z_i + z_j \le 1 + u_{ij} \end{aligned}\right\} \begin{array}{c}(\mathbf{z},\mathbf{u}) \in \\ \mathrm{LOCAL}(\mathcal{Q})\end{array} \\
& \mathbf{z},\mathbf{u} \quad \text{integer}\,.
\end{aligned}
$$

u_{ij} is 0 if either z_i or z_j are 0

u_{ij} is 1 if both z_i and z_j are 1

Solution Method 2

Introduce additional variables $u_{ij} = z_i z_j$ and constraints in order to form an equivalent integer linear program.

$$
\begin{aligned}
\min_{\mathbf{z},\mathbf{u}} \quad & \mathbf{c}^\top \mathbf{z} + \mathbf{q}^\top \mathbf{u} \\
& \mathbf{z} \in \mathrm{FLOW}_K \\
\text{s.t.} \quad & \left.\begin{array}{c} 0 \le u_{ij} \le 1, \ \forall ij \in \mathcal{Q} \\ u_{ij} \le z_i \,, \ u_{ij} \le z_j \\ z_i + z_j \le 1 + u_{ij} \end{array}\right\} \begin{array}{c} (\mathbf{z},\mathbf{u}) \in \\ \mathrm{LOCAL}(\mathcal{Q}) \end{array} \\
& \cancel{\mathbf{z}, \mathbf{u} \ \text{integer}} \,.
\end{aligned}
$$

This is then relaxed to a linear program (non-integer solns)

Rounding the Solution

To get back an integer 0,1 solution:

1) round the values → bad idea, may not satisfy FLOW

2) Hamming rounding – look for closest solution in FLOW
→ also not great, since that solution may not have a good objective function value.

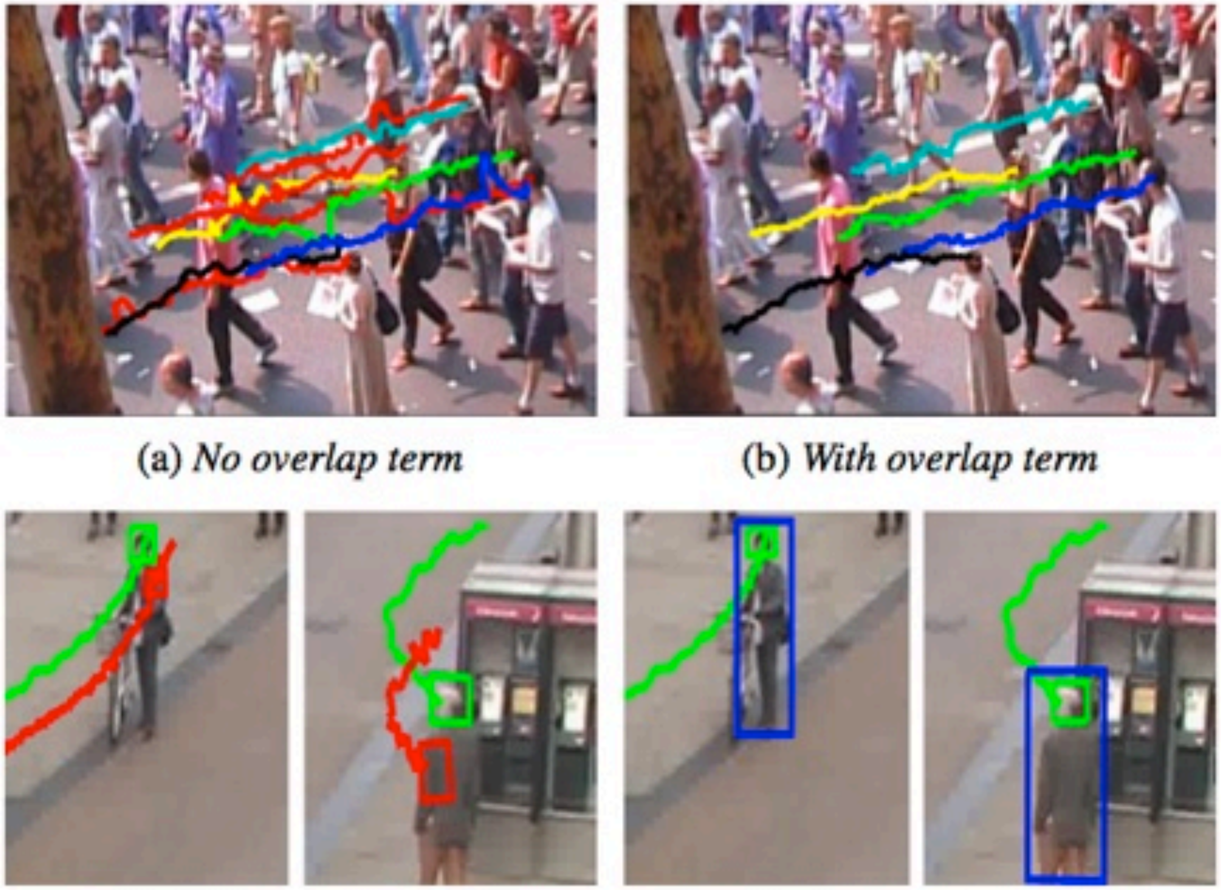
3) Frank-Wolfe rounding – one iteration of Frank-Wolfe algorithm by solving the linear program

$$\min_{\mathbf{z}} \quad (\mathbf{c} + (\mathbf{Q} + \mathbf{Q}^\top)\mathbf{z}^*)^\top \mathbf{z}$$
$$\text{s.t.} \quad \mathbf{z} \in \mathrm{FLOW}_K\,.$$

Bob's note: This Q is not convex, so this is only a heuristic.

Robert Collins
Penn State

Some Results



(a) *No overlap term* (b) *With overlap term*

(c) *No co-occurrence term* (d) *With co-occurrence term*

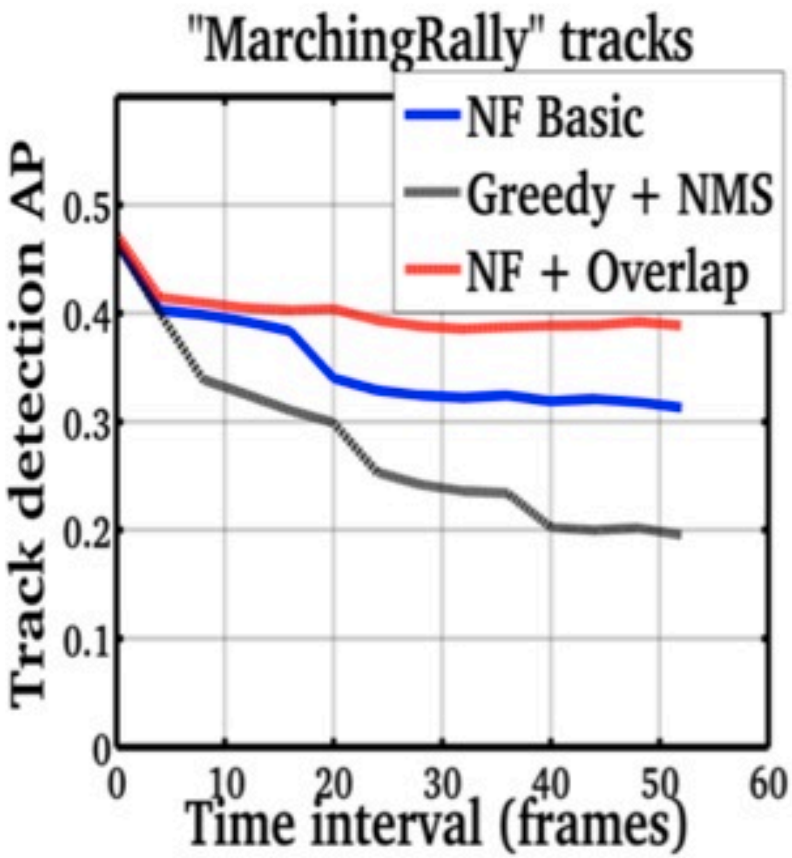
Figure 1: Results of network flow tracking using cost functions with/without pairwise terms. (a)-(b): a pairwise term that penalizes the overlap between different tracks helps resolving ambiguous tracks (shown in red) in crowded scenes. (c)-(d): a pairwise term that encourages the consistency between two signals (here head detections and body detections) helps eliminating failures (shown in red) of object detectors.

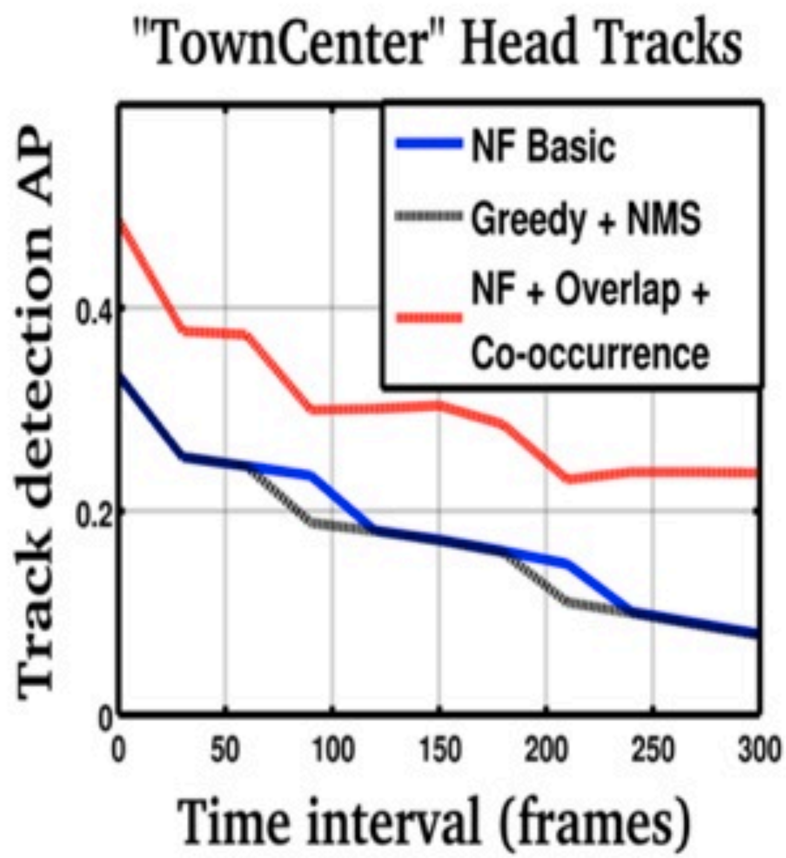
Re-detection Error

Re-detection measure. The proposed re-detection measure evaluates the ability of a tracker to find the correct location of a given object after Δt frames. The measure is inspired by the common evaluation procedure for object detection in still images [10] and extends it to tracking. For each pair of detections A_t and $B_{t+\Delta t}$ associated to the same track by a tracker, we check if there exists a ground truth track that overlaps with A_t and $B_{t+\Delta t}$ on frames t and $t + \Delta t$ respectively.[7] If the answer is negative, the subtrack $(A_t, B_{t+\Delta t})$ is labeled as false positive. Otherwise, it is labeled as true positive unless there exist multiple subtracks overlapping with the same ground truth. To avoid multiple responses, in the latter case only one subtrack is labeled as true positive while others are declared as false positives.

Pascal VOC

Some Results





Some Results

		Rcll	Prcn	GT	MT	PT	ML	FP	FN	IDs	FM	MOTA	MOTP
TUD Stadtmitte	NF	67.9	72.0	10	4	6	0	305	371	26	26	39.3	59.5
	NF+pairwise	59.6	**89.9**	10	2	8	0	**77**	467	15	22	51.6	**61.6**
	Milan [20]	**69.1**	85.6	10	**4**	**6**	0	134	**457**	15	**13**	**56.2**	**61.6**
PETS S2L1	NF	93.7	83.4	19	17	2	0	870	293	64	66	73.6	72.9
	NF+pairwise	92.4	**94.3**	19	**18**	**1**	0	**262**	354	56	74	85.5	**76.2**
	Milan [20]	**96.8**	94.1	19	18	**1**	0	282	**148**	**22**	**15**	**90.3**	74.3
PETS S2L2	NF	47.7	87.6	43	1	37	5	693	5383	291	531	38.1	60.7
	NF+pairwise	60.6	88.6	43	6	**34**	3	807	4050	244	379	50.4	**60.6**
	Milan [20]	**65.1**	**92.4**	43	**11**	31	**1**	**549**	**3592**	**167**	**153**	**58.1**	59.8
PETS S2L3	NF	44.5	92.2	44	9	15	20	164	2428	121	189	38.0	69.3
	NF+pairwise	**45.5**	91.2	44	**12**	15	**17**	155	**2125**	44	50	**40.3**	61.2
	Milan [20]	43.0	**94.2**	44	8	**17**	19	**115**	2493	**27**	**22**	39.8	**65.0**
PETS S1L1-2	NF	62.9	89.1	44	18	15	11	295	1425	289	140	47.8	65.2
	NF+pairwise	**68.9**	92.0	44	20	**16**	**8**	230	**1198**	35	74	**62.0**	**62.1**
	Milan [20]	64.9	**92.4**	44	**21**	12	11	**169**	1349	**22**	**19**	60.0	61.9
PETS S1L2-1	NF	31.3	87.4	42	4	15	23	208	3501	101	243	23.7	57.9
	NF+pairwise	**37.9**	89.6	42	**4**	**20**	**18**	223	**3141**	67	122	**32.2**	55.0
	Milan [20]	30.9	**98.3**	42	2	19	21	**27**	3494	**42**	**34**	29.6	**58.8**

Table 1: Table summarizing results over PETS and TUD sequences. Bold indicates best value for each column for each dataset. Abbreviations are as follows GT - ground truth tracks. MT - Mostly tracked. PT - partially tracked. ML - mostly lost. FP - false positives. FN - false negatives. IDs - ID swaps. FM - fragmentation.