Deep Network Flow for Multi-Object Tracking: Supplemental Material

Samuel Schulter Paul Vernaza Wongun Choi Manmohan Chandraker
NEC Laboratories America, Inc.
Cupertino, CA, USA

The supplemental material of our deep network flow approach for multi-object tracking contains the following items:

- Details on the formulation of deep network flows (Section 1)
- An on-line version of the tracker (Section 2)
- Qualitative results (Section 3)

1. Details on the formulation of deep network flows

First, we want to provide further details of our formulation of deep network flows and for computing gradients of the linear programming solution w.r.t. the cost functions.

1.1. The nullspace of $\mathbf{C}$ is larger than the trivial solution $\mathbf{0}$

In Section 3.2.1 of the main paper, we smooth the lower level problem, *i.e.*, the linear program. We get rid of the box constraints with log-barriers and remove the flow conservation constraints (matrix $\mathbf{C}$) with a change of basis, which requires the null space of $\mathbf{C}$. The matrix $\mathbf{C} \in \mathbb{R}^{2K \times M}$ models the flow conservation constraints

$$\begin{aligned} x_i^{\text{in}} + \sum_j x_{ji}^{\text{link}} &= x_i^{\text{det}} \\ x_i^{\text{out}} + \sum_j x_{ij}^{\text{link}} &= x_i^{\text{det}} \end{aligned} \tag{1}$$

for each detection $i = 1, \dots, K$. The dimensionality of the linear program is $M = 3K + |\mathcal{E}|$, where $\mathcal{E}$ is the set of all edges between detections, *i.e.*, x^{link}. The right singular vectors of $\mathbf{C}$ with corresponding 0 singular values define the null space of $\mathbf{C}$. The null space contains only the trivial solution $\mathbf{0}$, iff all columns of $\mathbf{C}$ are linearly independent. However, since the rank of $\mathbf{C}$ is at most $2K$, we have at least $K + |\mathcal{E}|$ singular vectors with a singular value of 0.

1.2. The bi-level formulation for computing gradients of the loss function w.r.t. the network flow costs

In Section 3.2.2 of the main paper, we directly use implicit differentiation on the optimality condition of the lower level problem to compute the gradients of the loss function $\mathcal{L}$ w.r.t. the costs $\mathbf{c}$ of the network flow problem, *i.e.*, $\frac{\partial \mathcal{L}}{\partial \mathbf{c}}$. For a more detailed derivation, we again define the bi-level problem from the main paper as

$$\mathcal{L}\left(\mathbf{x}(\mathbf{z}^*(\mathbf{c})), \mathbf{x}^{\text{gt}}\right) \text{ s.t. } \mathbf{z}^*(\mathbf{c}) = \arg\min_{\mathbf{z}} E(\mathbf{z}, \mathbf{c}) \tag{2}$$

with

$$E(\mathbf{z}, \mathbf{c}) = t \cdot \mathbf{c}^\top \mathbf{x}(\mathbf{z}) + P(\mathbf{x}(\mathbf{z})) \tag{3}$$

and

$$P(\mathbf{x}(\mathbf{z})) = -\sum_{i=1}^{2M} \log(\mathbf{b}_i - \mathbf{a}_i^\top \mathbf{x}(\mathbf{z})) \ . \tag{4}$$

For an uncluttered notation, we omit the dependency of $\mathbf{c}$ on $\mathbf{\Theta}$, which are the actual parameters of the cost functions to be learned. Note that computing $\frac{\partial \mathbf{c}}{\partial \mathbf{\Theta}}$ is essentially back-propagation of a neural network, which we use as cost functions, and

$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Theta}}$ is computed easily via the chain rule as $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Theta}} = \frac{\partial \mathbf{c}}{\partial \boldsymbol{\Theta}} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{c}}$ Using the optimality condition of the lower level problem (3) and Lagrange multipliers λ, we bring problem (2) into its unconstrained form

$$\mathcal{LG}(\mathbf{z}, \mathbf{c}, \lambda) = \mathcal{L}\left(\mathbf{x}(\mathbf{z}^*(\mathbf{c})), \mathbf{x}^{\text{gt}}\right) + \lambda^\top \cdot \frac{\partial E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{z}} \tag{5}$$

with new optimality conditions

$$\frac{\partial \mathcal{LG}(\mathbf{z}, \mathbf{c}, \lambda)}{\partial \mathbf{z}} = 0 = \frac{\partial \mathcal{L}}{\partial \mathbf{z}} + \frac{\partial^2 E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{z}^2} \cdot \lambda \tag{6}$$

$$\frac{\partial \mathcal{LG}(\mathbf{z}, \mathbf{c}, \lambda)}{\partial \mathbf{c}} = 0 = \frac{\partial^2 E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{c} \partial \mathbf{z}} \cdot \lambda \tag{7}$$

$$\frac{\partial \mathcal{LG}(\mathbf{z}, \mathbf{c}, \lambda)}{\partial \lambda} = 0 = \frac{\partial E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{z}} \tag{8}$$

The last optimality condition (8) is fulfilled by solving the linear program (LP), *i.e.*, the network flow. By using the first condition (6) we can compute the Lagrange multipliers as

$$\lambda = -\left[\frac{\partial^2 E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{z}^2}\right]^{-1} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{z}} = -\mathbf{H}_E^{-1} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{z}} \,, \tag{9}$$

where $\mathbf{H}_E$ is the Hessian of the lower level problem (3). Finally, we can define the gradients of the original problem w.r.t. the costs $\mathbf{c}$ as

$$\frac{\partial \mathcal{LG}(\mathbf{z}, \mathbf{c}, \lambda)}{\partial \mathbf{c}} = -\frac{\partial^2 E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{c} \partial \mathbf{z}} \cdot \mathbf{H}_E^{-1} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{z}} \,, \tag{10}$$

which is equivalent to the generic bi-level solution given in [3].

Finally, we define each of the three terms in (10) in more detail. Based on this first derivative of the lower level problem

$$\begin{aligned} \frac{\partial E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{z}} &= t \cdot \frac{\partial \mathbf{x}(\mathbf{z})}{\partial \mathbf{z}} \cdot \mathbf{c} + \frac{\partial \mathbf{x}(\mathbf{z})}{\partial \mathbf{z}} \frac{\partial P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})} \\ &= t \cdot \mathbf{B}^\top \cdot \mathbf{c} + \mathbf{B}^\top \frac{\partial P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})} \\ &= \mathbf{B}^\top \left[t \cdot \mathbf{c} + \frac{\partial P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})}\right] \,, \end{aligned} \tag{11}$$

we can define

$$\begin{aligned} \frac{\partial^2 E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{c} \partial \mathbf{z}} &= \left[\frac{\partial}{\partial \mathbf{c}} t \cdot \mathbf{c} + \frac{\partial}{\partial \mathbf{c}} \frac{\partial P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})}\right] \mathbf{B} \\ &= \left[t \cdot \mathbf{I} + \mathbf{0}\right] \mathbf{B} \\ &= t \cdot \mathbf{B} \,, \end{aligned} \tag{12}$$

$$\begin{aligned} \frac{\partial^2 E(\mathbf{z}, \mathbf{c})}{\partial \mathbf{z}^2} &= \left[\frac{\partial}{\partial \mathbf{z}} t \cdot \mathbf{c} + \frac{\partial}{\partial \mathbf{z}} \frac{\partial P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})}\right] \mathbf{B} \\ &= \left[\mathbf{0} + \frac{\partial \mathbf{x}(\mathbf{z})}{\partial \mathbf{z}} \frac{\partial^2 P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})^2}\right] \mathbf{B} \\ &= \mathbf{B}^\top \frac{\partial^2 P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})^2} \mathbf{B} \end{aligned} \tag{13}$$

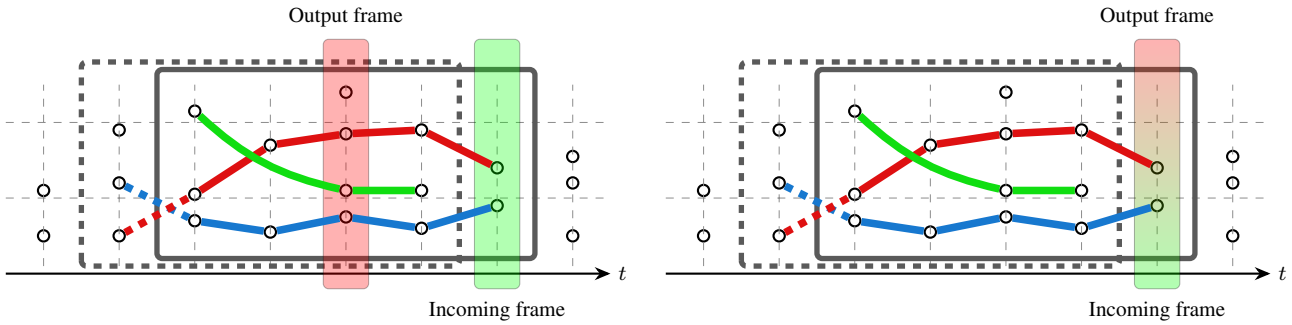
and

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}} = \frac{\partial \mathbf{x}(\mathbf{z})}{\partial \mathbf{z}} \cdot \frac{\partial \mathcal{L}}{\mathbf{x}(\mathbf{z})} = \mathbf{B}^\top \cdot \frac{\partial \mathcal{L}}{\mathbf{x}(\mathbf{z})} \,, \tag{14}$$

where $\mathbf{I}$ is the identity matrix. This gives the same solution as in the main paper, *i.e.*,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{c}} = -t \cdot \mathbf{B} \left[\mathbf{B}^\top \frac{\partial^2 P(\mathbf{x}(\mathbf{z}))}{\partial \mathbf{x}(\mathbf{z})^2} \mathbf{B}\right]^{-1} \mathbf{B}^\top \cdot \frac{\partial \mathcal{L}}{\mathbf{x}(\mathbf{z})} = -t \cdot \mathbf{B} \mathbf{H}_E^{-1} \mathbf{B}^\top \cdot \frac{\partial \mathcal{L}}{\mathbf{x}(\mathbf{z})} \,. \tag{15}$$

In the next section, we show that the Hessian $\mathbf{H}_E$ is always invertible.



(a) (b)

Figure 1: Illustration of (a) the off-line (or near on-line [1]) and (b) the on-line tracking model. For both figures, each node represents one detection at a time step t. Note that, for clarity we only use a single node to represent a detection instead of the actual two nodes (*c.f.*, Figure 1 in the main paper). In each figure the dark gray big rectangle represents the current temporal window (with a length of $W = 5$) we are solving the network flow problem for. The dashed version illustrates the previous temporal window. This also shows that we move the temporal window by one frame, *i.e.*, $\Delta = 1$. The green, blue and red curves show the three trajectories of the current solution. Again, the dashed curves illustrate (parts of) the trajectories from the previous solution. In the off-line version (a), the green transparent box marks the current incoming frame and the red transparent box marks that frame that is used to produce the output. For the on-line version (b), these two boxes overlap. The solution of the incoming frame is used as the output. The off-line (or near on-line [1]) version is typically more stable as it can look into both future and past frames.

1.3. The Hessian of the lower level problem is always invertible

The Hessian of the lower level problem (3) (again, with dependencies of functions omitted for an uncluttered notation) is given as

$$\mathbf{H}_E = \frac{\partial^2 E}{\partial \mathbf{z}^2} = \mathbf{B}^\top \cdot \frac{\partial^2 P}{\partial \mathbf{x}^2} \cdot \mathbf{B} = \mathbf{B}^\top \left[\sum_{i=1}^{2M} \frac{1}{\left(b_i - \mathbf{a}_i^\top \mathbf{x}\right)^2} \cdot \mathbf{a}_i \mathbf{a}_i^\top \right] \mathbf{B} \,, \tag{16}$$

where $\mathbf{a}_i^\top$ are the rows of $A = [\mathbf{I}, -\mathbf{I}]^\top$ (with $\mathbf{I}$ the identity matrix) and b_i are the values of the vector $\mathbf{b} = [\mathbf{1}, \mathbf{0}]^\top$.

Defining $\mathbf{e}_i$ as the unit vector with value 1 at dimension i and 0 elsewhere, we can see that $\mathbf{a}_i = \mathbf{e}_i$ for $i \leq M$, $\mathbf{a}_i = -\mathbf{e}_{i-M}$ for $i > M$, $b_i = 1$ for $i \leq M$ and $b_i = 0$ for $i > M$. Since $\mathbf{a}_i \mathbf{a}_i^\top = \mathbf{e}_i \mathbf{e}_i^\top = -\mathbf{e}_i \cdot -\mathbf{e}_i^\top$, we can write

$$\begin{aligned} \sum_{i=1}^{2M} \frac{1}{\left(b_i - \mathbf{a}_i^\top \mathbf{x}\right)^2} \cdot \mathbf{a}_i \mathbf{a}_i^\top &= \sum_{i=1}^{M} \left(1 - \mathbf{e}_i^\top \mathbf{x}\right)^{-2} \cdot \mathbf{e}_i \mathbf{e}_i^\top + \left(0 + \mathbf{e}_i^\top \mathbf{x}\right)^{-2} \cdot \mathbf{e}_i \mathbf{e}_i^\top \\ &= \sum_{i=1}^{M} \left((1 - x_i)^{-2} + x_i^{-2} \right) \cdot \mathbf{e}_i \mathbf{e}_i^\top \\ &= \operatorname{diag}\left[(1 - x_i)^{-2} + x_i^{-2} \right] = \mathbf{D} \,, \end{aligned} \tag{17}$$

where x_i is the value of $\mathbf{x}$ at dimension i and $\operatorname{diag}[\cdot]$ creates a diagonal matrix. Since we have $x_i \in (0, 1)$ because of the log barriers, all values of $\mathbf{D}$ are positive and finite and we can write the Hessian as

$$\mathbf{B}^\top \mathbf{D} \mathbf{B} = \mathbf{B}^\top \mathbf{D}^{\frac{1}{2}} \cdot \mathbf{D}^{\frac{1}{2}} \mathbf{B} = \hat{\mathbf{B}}^\top \hat{\mathbf{B}} \,. \tag{18}$$

The rank of $\mathbf{D}^{\frac{1}{2}} \in \mathbb{R}^{M \times M}$ is M and the rank of $\mathbf{B} \in \mathbb{R}^{M \times L}$ is $L = K + |\mathcal{E}|$. Via matrix rank properties (*e.g.*, Sylvester's rank inequality), $\hat{\mathbf{B}}$ and also its Gram matrix have rank L, which means the Hessian $\frac{\partial^2 E}{\partial \mathbf{z}^2} \in \mathbb{R}^{L \times L}$ has full rank.

2. On-line tracking

As noted in Section 3.4 of the main paper, our tracking model can process a video sequence on-line, *i.e.*, without taking future frames into account. Figure 1 illustrates this causal system. We did one experiment on the KITTI-Tracking data set [2]

	MOTA	REC	PREC	MT	IDS	FRAG
MLP 2	74.19	84.07	92.85	59.96	70	376
MLP 2 (online)	72.34	81.37	93.68	49.74	69	351

Table 1: Off-line (or near on-line [1]) versus on-line processing results on a cross-validation on KITTI-Tracking [2].

to compare the on-line version with our off-line (or near on-line [1]) version. Based on the same experimental setup as in Section 4.1 of the main paper, we compare the off-line and on-line version of the MLP 2 model. Table 1 demonstrates that the on-line version of our tracking model only shows a moderate drop in performance and mainly affects the recall (REC and MT). However, the on-line version enables many applications that require strict on-line processing of streaming data, *e.g.*, autonomous driving.

3. Qualitative results

In this section, we present some qualitative results, which compare two different models in each example. In each of the figures, we show the output of two models side-by-side for different frames, *i.e.*, the first frame is in the first row and the last frame is in the last row for each model. The green dotted bounding boxes represent the ground truth. Solid boxes are the detections, where detections from the same trajectory have the same color and the same number (object ID) on top of the corresponding bound box. The background color of the object ID indicates if the corresponding detection is a true (green background) or false positive (red background).

Figures 2 to 6 compare models with hand-crafted costs (left) and learned costs (right). The learned model has a 2-layer MLP that processes only bounding box information. Figures 7 to 10 compare different types of input for learned cost functions. The model in the left figures only sees bounding box information (*c.f.* (B+O) in the paper), whereas the model on the right also uses RGB patches for the unary potential, (*c.f.* Au+(B+O) in the paper). Finally, we show some negative examples in Figures 11 to 13.

References

[1] W. Choi. Near-Online Multi-target Tracking with Aggregated Local Flow Descriptor. In *ICCV*, 2015. 3, 4

[2] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *CVPR*, 2012. 3, 4

[3] P. Ochs, R. Ranftl, T. Brox, and T. Pock. Bilevel Optimization with Nonsmooth Lower Level Problems. In *SSVM*, 2015. 2

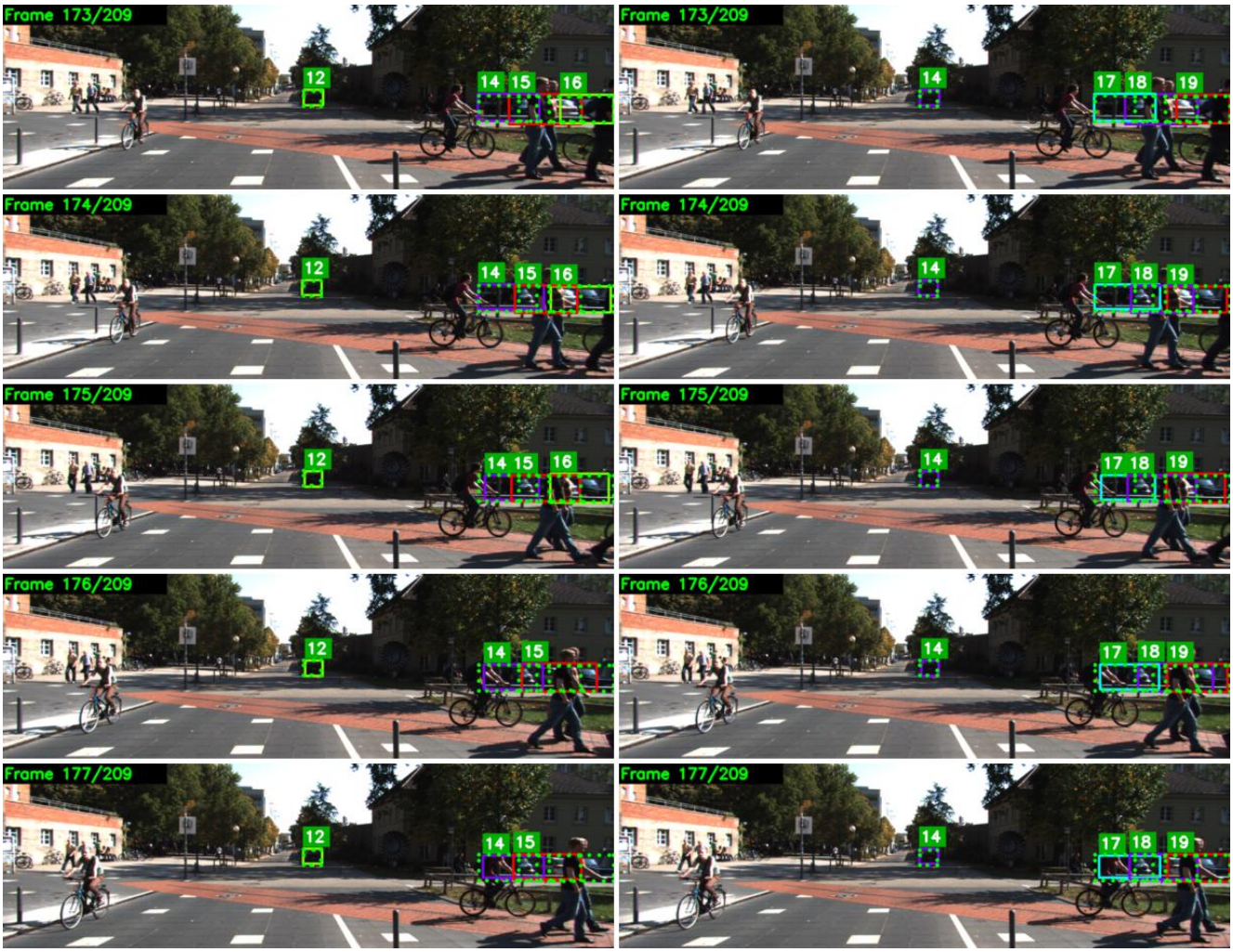


Figure 2: Left: model with hand-crafted costs. Right: model with learned costs from bounding box information. Note the failure case of the hand-crafted model in the last two frames in the right part of the image. Only two of the three cars are correctly tracked.

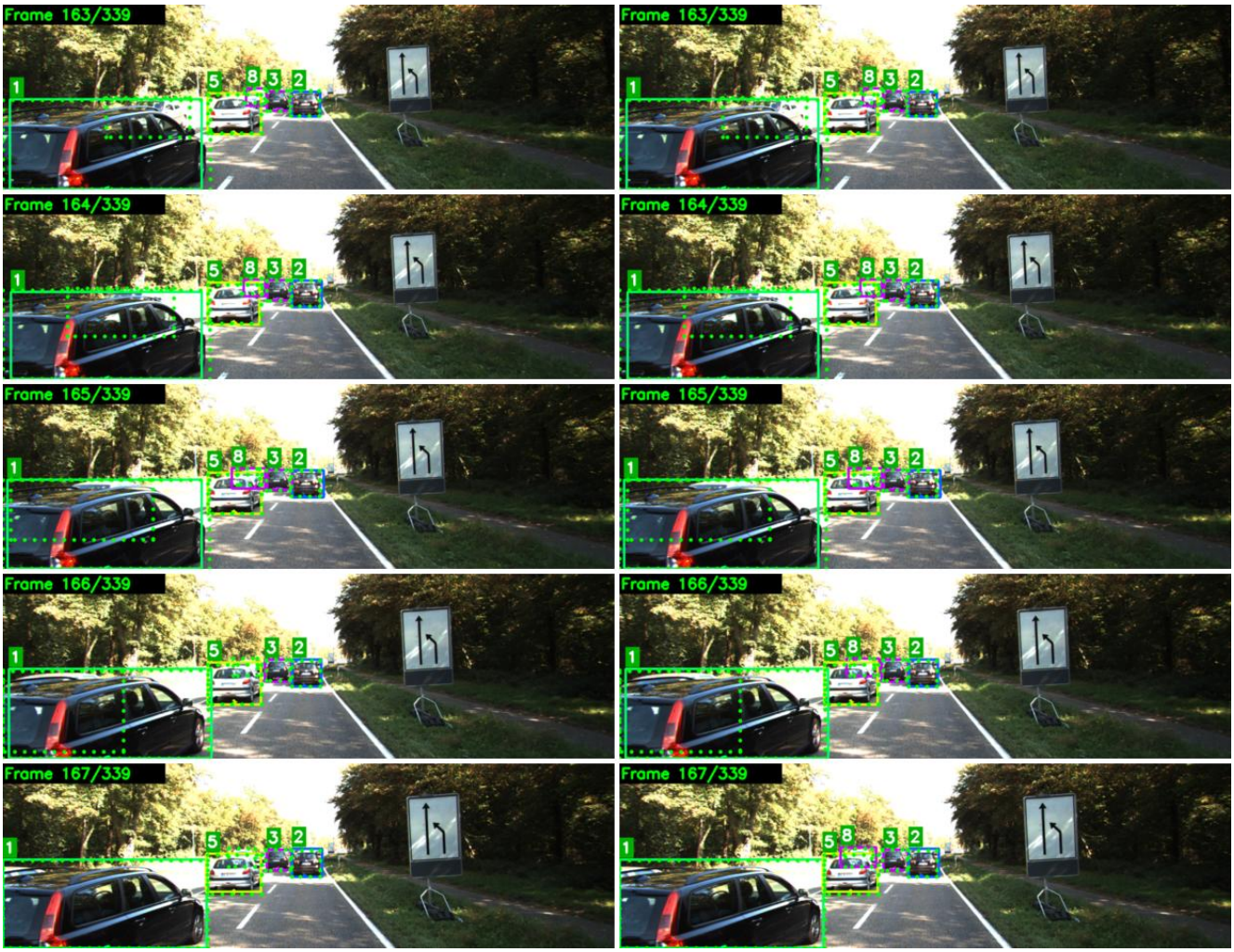


Figure 3: Left: model with hand-crafted costs. Right: model with learned costs from bounding box information. Note the failure case of the hand-crafted model in the last two frames in the middle part of the image. The car with the ID 8 is not tracked anymore.

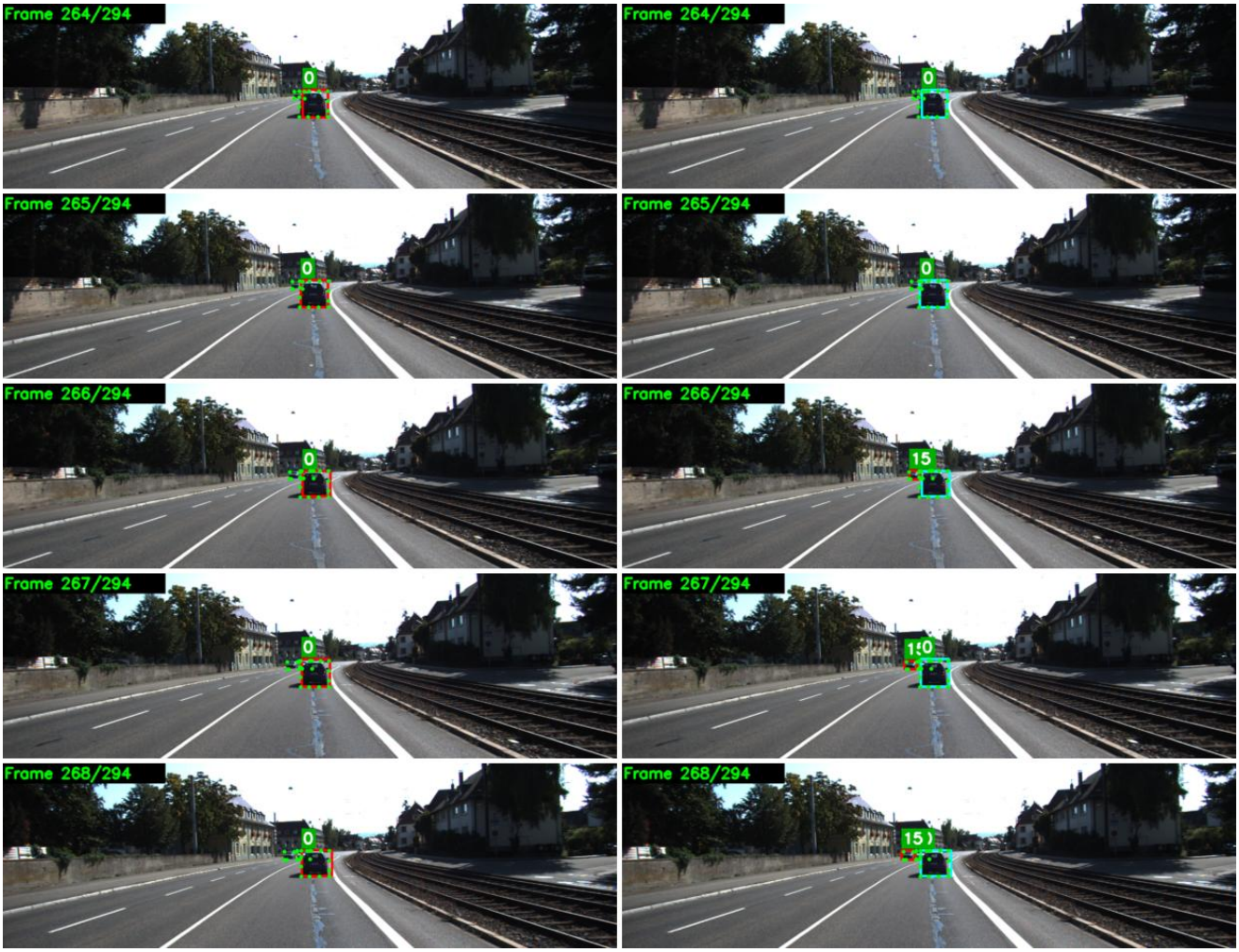


Figure 4: Left: model with hand-crafted costs. Right: model with learned costs from bounding box information. The hand-crafted model is not able to detect the second car at all. The model with learned costs successfully tracks the car from the third frame onwards.

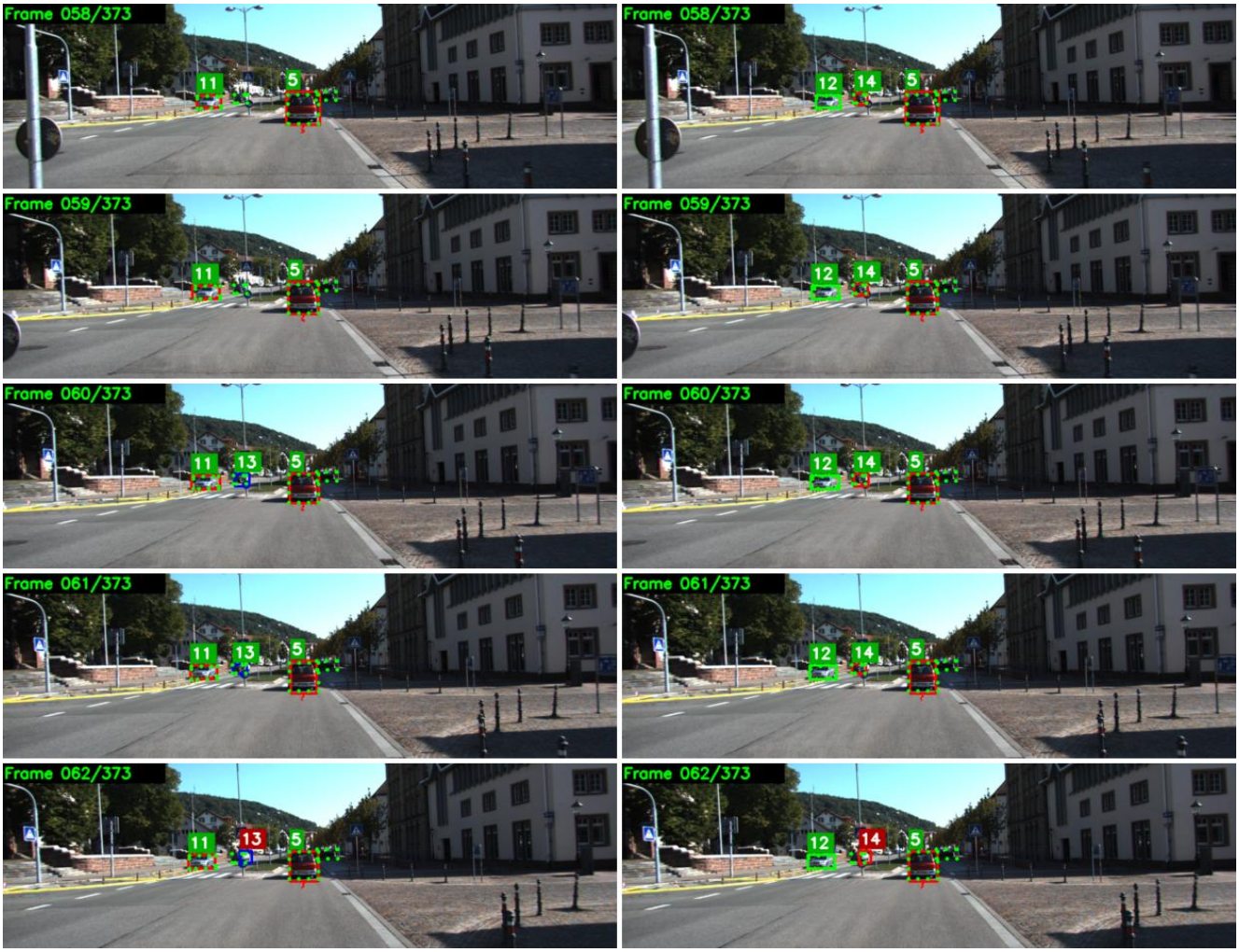


Figure 5: Left: model with hand-crafted costs. Right: model with learned costs from bounding box information. While both models have a false positive detection in the last frame due to inaccurate localization, the model with hand-crafted costs cannot track that car in the first two frames.

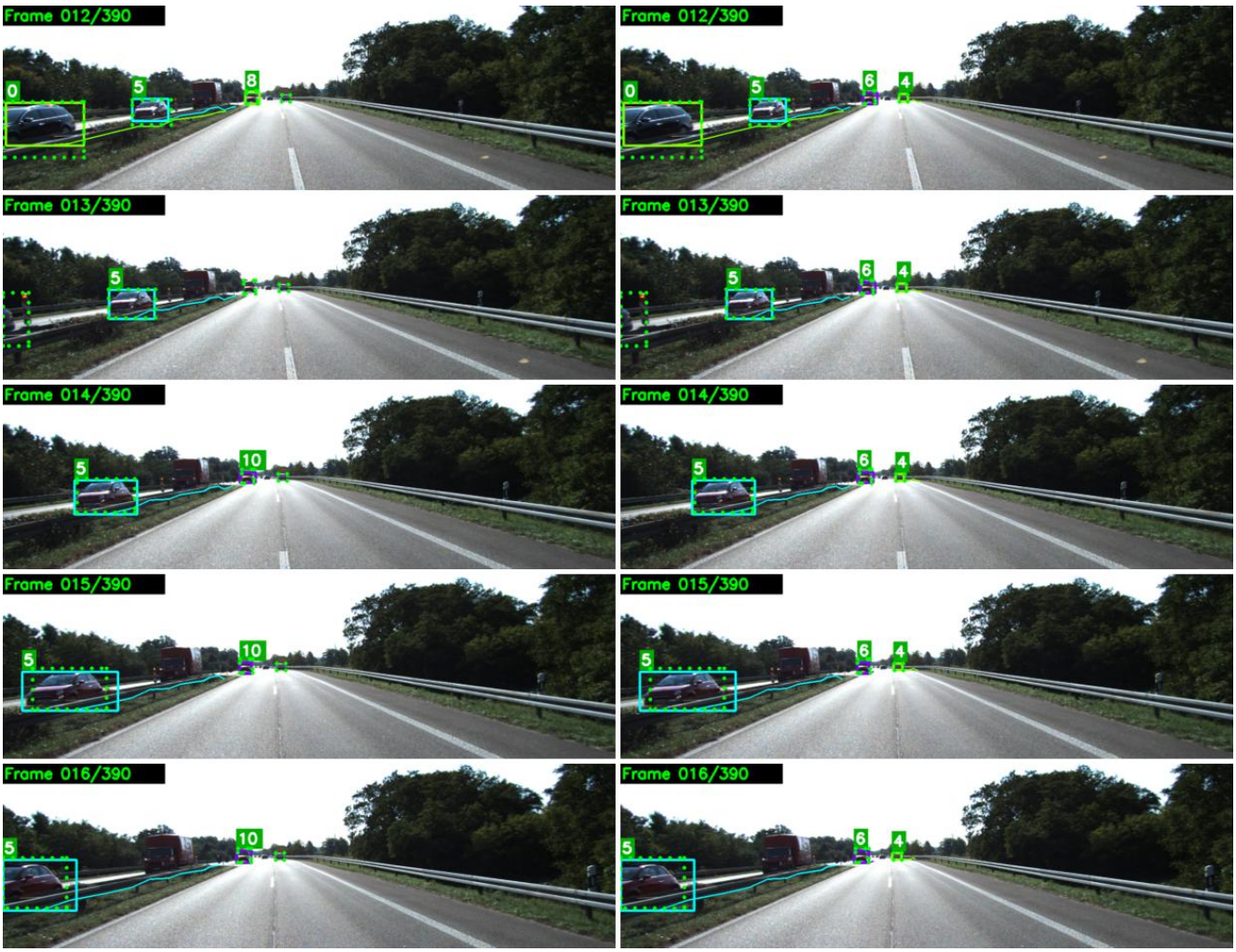


Figure 6: Left: model with hand-crafted costs. Right: model with learned costs from bounding box information. The model with hand-crafted costs has problems detecting and tracking the two small cars in the model of the image.

Figure 7: Left: model with only bounding box information as input for learning cost functions. Right: model with appearance features as input (raw RGB patches) for the unary term. Having access to appearance features seems to help in suppressing false positives from the underlying detector.

Figure 8: Left: model with only bounding box information as input for learning cost functions. Right: model with appearance features as input (raw RGB patches) for the unary term. Having access to appearance features seems to help in suppressing false positives from the underlying detector.

Figure 9: Left: model with only bounding box information as input for learning cost functions. Right: model with appearance features as input (raw RGB patches) for the unary term. The model using appearance features detects one more person on the left side of the image, while the other model misses it.

Figure 10: Left: model with only bounding box information as input for learning cost functions. Right: model with appearance features as input (raw RGB patches) for the unary term. The model using appearance features detects two more person (object IDs 28 and 25), while the other model misses them.

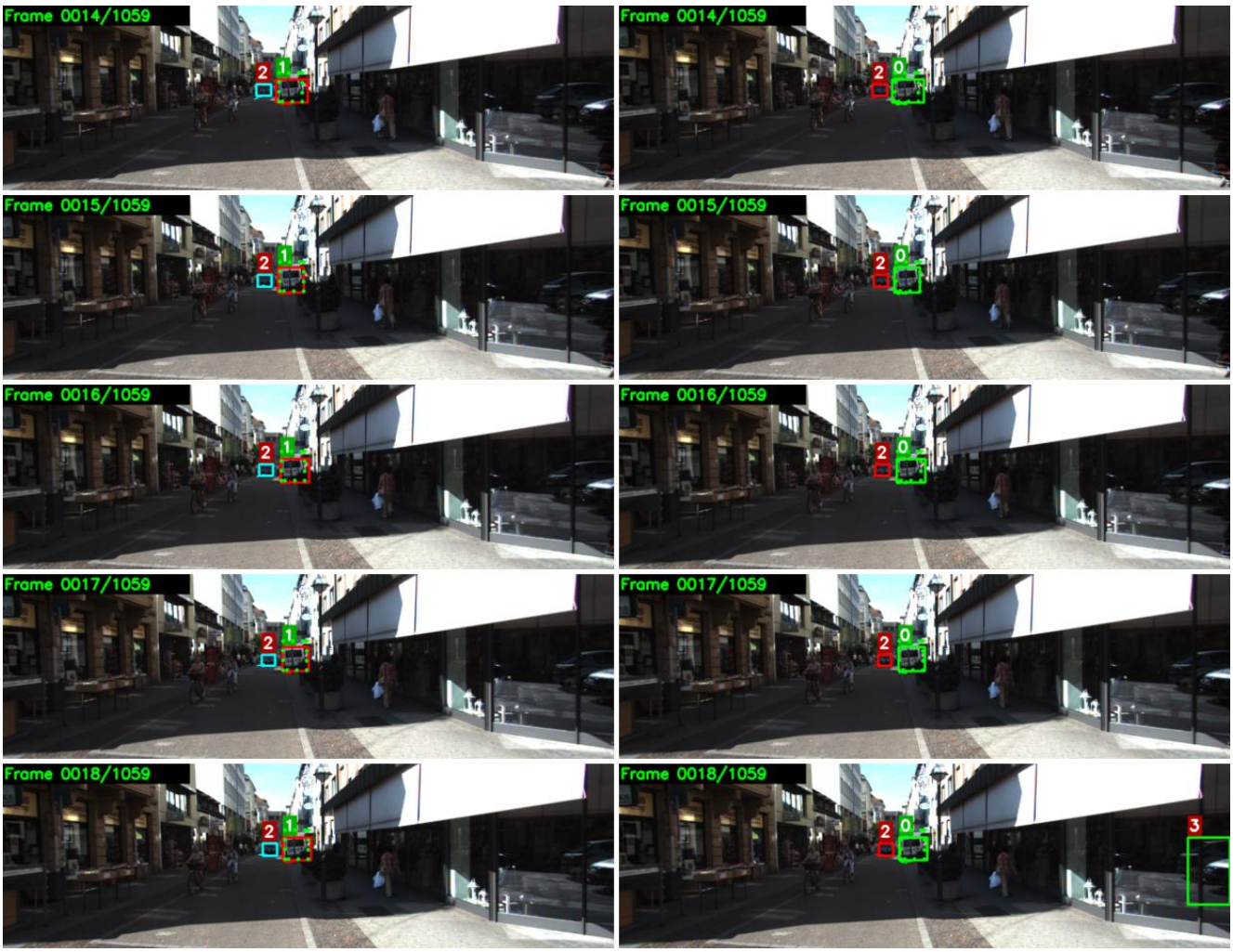


Figure 11: Left: model with hand-crafted costs. Right: model with learned costs from bounding box information. In this failure case, the model with learned costs has one more false positive in the last frame on the right side of the image.

Figure 12: Left: model with hand-crafted costs. Right: model with learned costs from bounding box information. In this failure case, the model with learned costs has one more false positive in the last frame on the left side of the image.

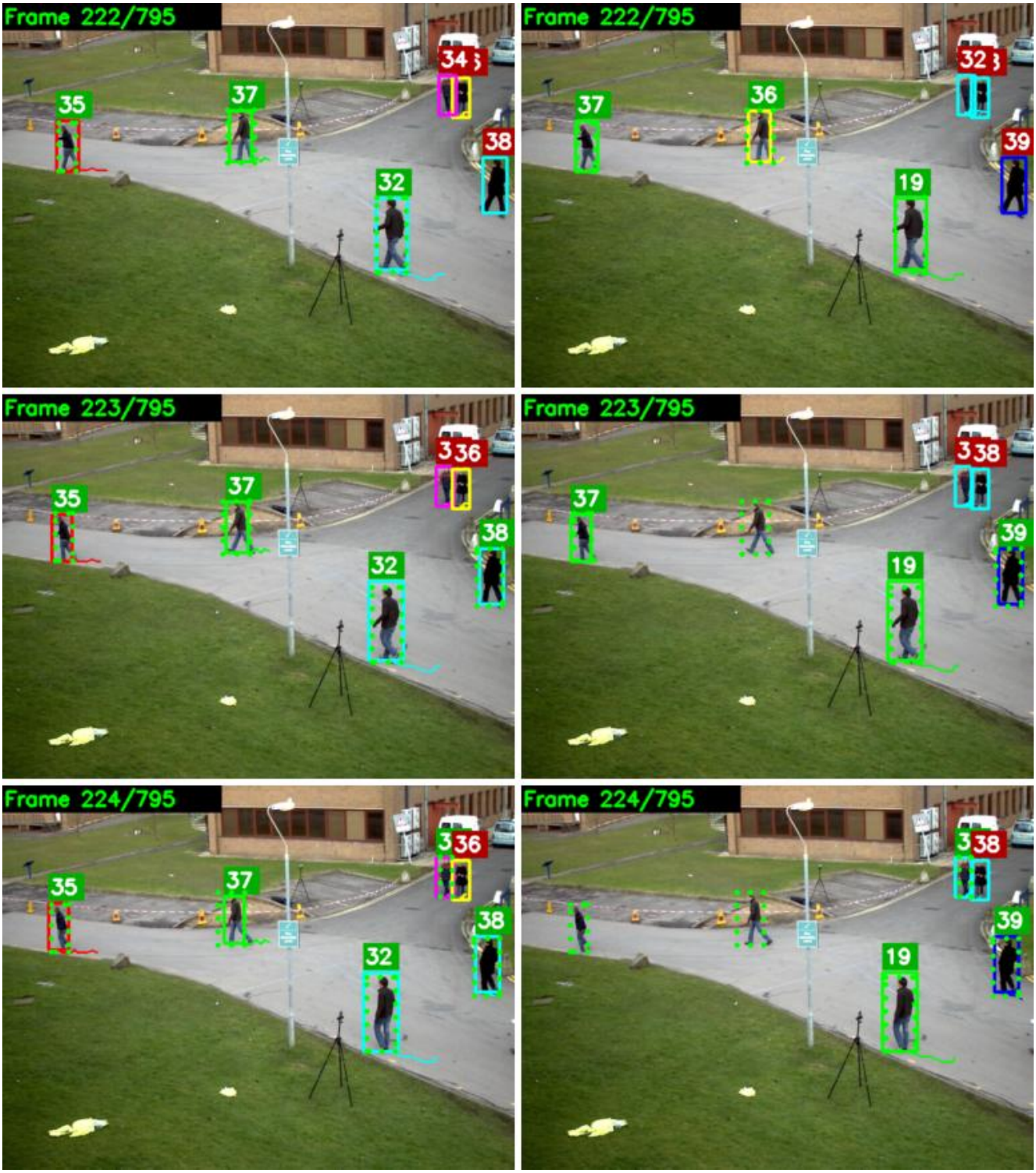


Figure 13: Left: model with only bounding box information as input for learning cost functions. Right: model with appearance features as input (raw RGB patches) for the unary term. In this failure case, the model using appearance features does not correctly track two pedestrians in the last two frames (object IDs 36 and 37).