3D Object Tracking and Localization for AI City

Gaoang Wang, Zheng Tang, Jenq-Neng Hwang

Information processing lab, University of Washington
Success of CNN Vehicle Detectors (YOLOv2\textsuperscript{[1]})

- Where are the cars in world coordinates?
- What is the GPS speed of each car?

Challenges of Tracking by Detection

- Noisy Detection
- Occlusion
- Appearance Change

Challenges
Tracklet-based Clustering

Input Video

Build Tracklets

Appearance

t1-t4

t6-t10

t7-t11

Trajectory
Adaptive Appearance Modeling

• Histogram-based adaptive appearance model
  • A history of spatially weighted (kernel) histogram combinations will be kept for each vehicle

The first row respectively presents the RGB, HSV, Lab, LBP and gradient feature maps for an object instance in a tracklet, which are used to build feature histograms.
The second row shows the original RGB color histograms.
The third row demonstrates the Gaussian spatially weighted (kernel) histograms, where the contribution of background area is suppressed.
Tracklet-based Clustering

• Clustering Loss

\[ l = \lambda_{sm}l_{sm} + \lambda_{ac}l_{ac} + \lambda_{ti}l_{ti} \]

- Smoothness in the trajectory
- Appearance change
- How far away in the time domain

**Loss?**

**Same trajectory**

**Different trajectory**

**Black dots** show the detected locations at time \( t \).

**Red curves** represent trajectories from Gaussian regression.

**Green dots** show \( n_k \) neighboring points on the red curves around the endpoints of the tracklets at \( t_{j,nd} \) and \( t_{j+1,st} \).
Tracklet-based Clustering

• **Edge** represents the clustering loss between two nodes (tracklets).
Optimization by Clustering

- **A)** Assign
  - Denote the trajectory set of the $j$-th tracklet as $S(j)$ which is a set of tracklets belonging to the trajectory. The loss change after assign operation can be expressed by,

  $$
  \Delta l_a = \min_i (l(S(j) \setminus \{j\}) + l(S_i \cup \{j\})) - (l(S(j)) + l(S_i))
  $$

  Loss after operation
  Loss before operation

  ![Diagram](image)

  - Cluster $S(j)$
  - Cluster $S(i)$

  before
  after
Optimization by Clustering

• B) Merge

\[ \Delta l_m = \min_i \left( l(S(j) \cup S_i) \right) - \left( l(S(j)) + l(S_i) \right) \]
Optimization by Clustering

• C) Split

\[ \Delta l_{sp} = (l(\{j\}) + l(S(j) \setminus \{j\})) - l(S(j)) \]

Cluster \( S(j) \)

Cluster \( S(i) \)

before

after
Optimization by Clustering

D) **Switch**

For the $j$-th tracklet, denote all the tracklets in $S(j)$ after the $j$-th tracklet as $S_{aft}(j)$ and other tracklets as $S_{bef}(j)$. Then make the same splitting for all the trajectory set based on the $j$-th tracklet. Then we switch $S_{bef}(j)$ and $S_{i,bef}$ to calculate the loss change as follows,

$$
\Delta l_{sw} = \min_i \left( l(S_{bef}(j) \cup S_{i,aft}) + l(S_{aft}(j) \cup S_{i,bef}) \right) - \left( l(S(j)) + l(S_i) \right)
$$
Optimization by Clustering

• E) Break

\[ \Delta l_b = \left( l(S_{bef}(j)) + l(S_{aft}(j)) \right) - l(S(j)) \]
Resulting Trajectories from Tracklets
Camera Calibration

• Minimization of reprojection error solved by EDA

\[
\min_{\mathbf{P}} \sum_{k=1}^{N_{1s}} \left| \| \mathbf{P}_k - \mathbf{Q}_k \|_2 - \| \hat{\mathbf{P}}_k - \hat{\mathbf{Q}}_k \|_2 \right|
\]

s. t. $\mathbf{P} \in \text{Rng}_\mathbf{P}, \mathbf{p}_k = \mathbf{P} \cdot \hat{\mathbf{P}}_k, \mathbf{q}_k = \mathbf{P} \cdot \hat{\mathbf{Q}}_k$

\( \mathbf{P} \): Camera projection matrix
\( \text{Rng}_\mathbf{P} \): Range for optimization
\( \mathbf{P}_k, \mathbf{Q}_k \): True endpoints of line segments
\( \hat{\mathbf{P}}_k, \hat{\mathbf{Q}}_k \): Estimated endpoints of line segments
\( \mathbf{p}_k, \mathbf{q}_k \): 2D endpoints of line segments
\( N_{1s} \): Number of endpoints
Results on AI City Challenge 2018 (Track 1) \cite{1,2}

- Track 1 - Traffic flow analysis
  - 27 videos, each 1 minute in length, recorded at 30 fps and 1080p resolution
  - Performance evaluation: \( S_1 = DR \times (1 - NRMSE) \)
  - \( DR \) is the detection rate and \( NRMSE \) is the normalized Root Mean Square Error (RMSE) of speed
- 56 teams participated, 13 teams submitted the final results.

\begin{thebibliography}{9}
\bibitem{1} Naphade, M., Chang, M. C., Sharma, A., Anastasiu, D. C., Jagarlamudi, V., Chakraborty, P., ... & Hwang, J. N. (2018). The 2018 NVIDIA AI City Challenge. In \textit{Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops} (pp. 53-60).
\end{thebibliography}
Results on AI City Challenge 2018 (Track 1)

Acknowledgement
We thank NVIDIA for organizing AI City Challenge and providing the dataset for training and evaluation.

DR: 1.0000    RMSE: 4.0963 mi/h
General Multi-Object Tracking (Ongoing)
Connectivity Loss

• The loss for merging two tracklets
  • Same ID
  • Different ID

  ![Diagram showing connectivity loss]

  Similarity \( \approx 1 \)

  Similarity \( \approx 0 \)
TrackletNet
TrackletNet

- Input tensor \((B \times D \times T \times C)\)
  - \(B\) (32): batch size in the training.
  - \(D\) (516): feature dimension for each detection.
    - 4-D Location feature: \(x, y, \text{width}, \text{height}\).
    - 512-D appearance feature: learned from FaceNet [1].
  - \(T\) (64): time window
  - \(C\) (3): input channels
    - \(C_1\): two embedded tracklet feature map.
    - \(C_2\): binary mask to indicate the location of 1\(^{st}\) tracklet.
    - \(C_3\): binary mask to indicate the location of 2\(^{nd}\) tracklet.

TrackletNet

• Architecture
  • **4 sizes of convolution kernels:** 1 x 3, 1 x 5, 1 x 9, 1 x 13. Different kernels can deal with different lengths of missing detections.
  • **3 convolution and max pooling layers:** feature extraction.
  • **1 average pooling** on appearance dimensions after the last max pooling: weighted majority vote on 512 dimensions of appearance features to measure the appearance change.
  • **2 fully connected layers.**
Properties of TrackletNet

• Convolution along time domain only.
  • No convolution across feature space.
  • The complexity of the network is largely reduced, which can **address overfitting**.

• Convolution solves **connectivity** loss.
  • Convolution includes **lowpass** and **highpass** filters in time domain.
  • Lowpass filters can suppress the detection noise.
  • Highpass filters can measure whether there are abrupt changes.

• Binary masks can tell the missing detections to the network.
Convert to 3D Tracking

• $(x_{2d}, y_{2d}, w_{2d}, h_{2d}) \rightarrow (x_{3d}, y_{3d}, z_{3d}, w_{3d}, h_{3d})$

• Obtain 3D
  • Estimate foot location $(x_{3d}, y_{3d}, z_{3d}) = (X_2 + X_1)/2$ by ground plane.
  • $w_{3d} = ||X_2 - X_1||$, $h_{3d} = w_{3d} \times h_{2d} / w_{2d}$
  • Detection location $(x_{3d}, z_{3d}, w_{3d}, h_{3d})$. Drop $y_{3d}$ out since $(x_{3d}, y_{3d}, z_{3d})$ are linear dependent.
TrackletNet Training (2D and 3D)

• The same input size of 2D and 3D tracking. They share the same architecture.

• Augmentation in training
  • Bounding box randomization
    • Randomly disturb the size and location of bounding boxes by a factor of random noise sampled from normal distribution with mean and standard deviation to be 0 and 0.05, respectively.
  • Tracklet random split
    • Randomly divide each trajectory into small pieces of tracklets.
  • Tracklet random combination
    • Randomly select two tracklets as the input of the network.
MOT Challenge 2016\textsuperscript{[1]}

## Results on MOT Benchmark

<table>
<thead>
<tr>
<th>Tracker</th>
<th>IDF1</th>
<th>MOTA</th>
<th>MT%</th>
<th>ML%</th>
<th>FP</th>
<th>FN</th>
<th>ID sw</th>
<th>Frag</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ours</strong></td>
<td>56.1</td>
<td>49.2</td>
<td>17.3</td>
<td>40.3</td>
<td>8,400</td>
<td>83,702</td>
<td>606</td>
<td>882</td>
</tr>
<tr>
<td>TLMHT</td>
<td>55.3</td>
<td>48.7</td>
<td>15.7</td>
<td>44.5</td>
<td>6,632</td>
<td>86,504</td>
<td>413</td>
<td>642</td>
</tr>
<tr>
<td>DMMOT</td>
<td>54.8</td>
<td>46.1</td>
<td>17.4</td>
<td>42.7</td>
<td>7,909</td>
<td>89,874</td>
<td>532</td>
<td>1,616</td>
</tr>
<tr>
<td>NOMT</td>
<td>53.3</td>
<td>46.4</td>
<td>18.3</td>
<td>41.4</td>
<td>9,753</td>
<td>87,565</td>
<td>359</td>
<td>504</td>
</tr>
<tr>
<td>eHAF16</td>
<td>52.4</td>
<td>47.2</td>
<td>18.6</td>
<td>42.8</td>
<td>12,586</td>
<td>83,107</td>
<td>542</td>
<td>787</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tracker</th>
<th>IDF1</th>
<th>MOTA</th>
<th>MT%</th>
<th>ML%</th>
<th>FP</th>
<th>FN</th>
<th>ID sw</th>
<th>Frag</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ours</strong></td>
<td>58.0</td>
<td>51.9</td>
<td>23.5</td>
<td>35.5</td>
<td>37,311</td>
<td>231,658</td>
<td>2,294</td>
<td>2,917</td>
</tr>
<tr>
<td>TLMHT</td>
<td>56.5</td>
<td>50.6</td>
<td>17.6</td>
<td>43.4</td>
<td>22,213</td>
<td>255,030</td>
<td>1,407</td>
<td>2,079</td>
</tr>
<tr>
<td>DMMOT</td>
<td>55.7</td>
<td>48.2</td>
<td>19.3</td>
<td>38.3</td>
<td>26,218</td>
<td>263,608</td>
<td>2,194</td>
<td>5,378</td>
</tr>
<tr>
<td>NOMT</td>
<td>54.7</td>
<td>51.8</td>
<td>23.4</td>
<td>37.9</td>
<td>33,212</td>
<td>236,772</td>
<td>1,834</td>
<td>2,739</td>
</tr>
<tr>
<td>jCC</td>
<td>54.5</td>
<td>51.2</td>
<td>20.9</td>
<td>37.0</td>
<td>25,937</td>
<td>247,822</td>
<td>1,802</td>
<td>2,984</td>
</tr>
</tbody>
</table>

**MOT16**

**MOT17**
Examples and Applications

• 1. 3D Pose estimation
  • Not many works related to multi-person pose estimation.
  • Not many works dealing with missing pose and occlusions.
  • Tracking can be treated as a preprocessing step to the above issues.
Examples and Applications

• Use pre-trained model on MOT without finetune.
• Detection: OpenPose
Examples and Applications

• 2. Autonomous Driving
  • Estimate the speed of pedestrian.
  • Anomaly detection of person behaviors.
  • Tracking can be also involved in ground plane estimation and bundle adjustment as additional constraints.
Examples and Applications

- Detection: Yolo
Examples and Applications

• 3. Drone applications
  • Similar to autonomous driving.
  • Use pre-trained model on KITTI without finetune.
• Detection: Mask-RCNN