Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects

Hamed Pirsiavash, Deva Ramanan, Charless Fowlkes

Department of Computer Science, UC Irvine
Estimate number of tracks and their extent

- Do not initialize manually
- Estimate birth and death of each track
Our approach: Graph theoretic problem

- **Globally Optimal**
  - for a common class of objective functions
- **Locally Greedy**
  - and hence straightforward to implement
- **Scale linearly** in the number of objects and video length
• Object state \( x_t \in S \)

• Object track

\[
E(x) = \sum_{t} \text{Local}(x_t) + \text{Pair}(x_t, x_{t-1})
\]

(e.g. HMM)

• K-object tracker

\[
\arg\min_{x_1 \ldots x_K} \sum_{k} E(x_k)
\]

• Discretize state space \( S \) (e.g., scanning window locations)
• Assume no tracks overlap (for now)
• Must infer \( K \), track births & deaths, and solve data association
Trellis graph

- Local cost of window
- Pairwise cost of transition
- Dynamic programming finds a single track
  - (Viterbi algorithm)
Trellis graph

Add edges to model occlusion

- **Local cost** of window
- **Pairwise cost** of transition
- Dynamic programming finds a single track
  - (Viterbi algorithm)
What about variable length tracks?

How to find more than one track?
Equivalent graph problem: **Min-cost-flow**

A generalization of min-cut/max-flow problem

Introduced in “Zhang, Li, Nevatia, CVPR’08”
Our contribution

Find 4-track solution given a 3-track solution

3-track estimate

4-track estimate

DP
Our contribution

Find 4-track solution given a 3-track solution

3-track estimate

4-track estimate

Sub-optimum

DP

4-track estimate

Optimum
Our contribution

Find 4-track solution given a 3-track solution

3-track estimate

4-track estimate

Sub-optimum

SSP

Optimum

DP

Shortest path: New track can “suck” flow from existing tracks
Solutions

• Globally optimum
  – Previous work
    • Zhang et al CVPR’08: Introduced the model with a naïve solver \( O(N^3 \log^2 N) \)

  – Our algorithm
    • Exploits the special structure of graph (DAG, unit-capacity)
    • Is greedy using successive shortest path \( O(KN \log N) \)

• Approximate
  – Dynamic programming
    • Is greedy \( O(KN) \)
Why is greedy nice?

Non-max-suppression in the loop:
- At each iteration, suppress all windows overlapping with the instanced track.

One iteration
Experiments

Datasets:

- Caltech pedestrian dataset
  - Camera on a moving car
  - ~120,000 frames

- ETHMS dataset
  - Moving camera on a cross walk
  - ~1000 frames
Detection vs false positive per frame (FFPI) for ETHMS dataset
Detection vs false positive per frame (FFPI) for ETHMS dataset
Detection vs false positive per frame (FFPI) for ETHMS dataset
Novel, scalable, greedy algorithm with state-of-the-art results

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Detection rate</th>
<th>False positive per frame</th>
<th>Run time per frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stereo algorithm in Ess et al CVPR’08</td>
<td>47</td>
<td>1.5</td>
<td></td>
</tr>
<tr>
<td>Algorithm 1 in Zhang et al CVPR’08</td>
<td>68.3</td>
<td>0.85</td>
<td>95ms</td>
</tr>
<tr>
<td>Algorithm 2 with occlusion handling in Zhang et al CVPR’08</td>
<td>70.4</td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td>Xing et al</td>
<td>75.2</td>
<td>0.939</td>
<td></td>
</tr>
<tr>
<td>Our DP</td>
<td>76.6</td>
<td>0.85</td>
<td>0.5ms</td>
</tr>
<tr>
<td>Our DP+NMS</td>
<td>79.8</td>
<td>0.85</td>
<td>0.7ms</td>
</tr>
</tbody>
</table>
Thanks