Optimizing Pedestrian Detection for Real-time Automotive Applications
Khanh Vo Duc, Mobile Vision Team, NVIDIA
Agenda

- Introduction to Pedestrian Detection
- Survey of Pedestrian Detection Techniques
- Academic Focus
- Production ADAS Focus
- Optimizations
- Demo Video
Introduction to Pedestrian Detection

- Examples

Classification State-of-the-Art Techniques

- Haar wavelets
- Histograms of Oriented Gradients (HOG)
- Local Binary Patterns (LBP)
Classification State-of-the-Art Techniques

- Current state-of-the-art is:
  - Sped up version of *ChnFtrs*\(^1\) classifier (Integral Channel Filters)
  - HOG-like features + Cascade structure + SVM\(^2\)

\(^1\) Dollar et al - “Fastest pedestrian detector in the west”

\(^2\) Benenson et al - “Pedestrian detection at 100 frames per second”
Academic Focus

- Accuracy
- Novelty
- Trade-offs
- Grayscale / color
- Depth / stereo
- Public datasets
  - Not necessarily representative of in-vehicle camera footage (INRIA, TUD Brussels)
Production ADAS Focus

- In-vehicle integration
- Real-time operation
- Accuracy
- No false positives
  - Can put driver in dangerous situations
  - Reduces driver confidence in the system
- Cost
  - ASICs, FPGAs, General-purpose processors
- Power
From Academia to Production

- **Cameras**
  - Color vs. Grayscale
    - Implications of grayscale: Retraining classifiers, reduced detection rate
  - Monocular vs. Stereo
    - Implications of mono: No depth information, larger search space
  - Infrared
    - Can significantly simplify night vision detection

- **Sensor fusion**
  - Availability of lidar / radar
    - Can provide depth information for monocular cameras
Proposed Solution: Motion Estimation

- Observation: To an observer on a moving vehicle, closer objects move faster than objects farther away.
Towards ADAS: Motion Estimation

- Calculate motion vector for each $N \times N$ pixel block
  - Compute motion from previous frame to current frame
  - Possible methods:
    - Optical flow algorithms
    - Lucas-Kanade, Block-matching, Horn and Schunck
    - Block-based Iterative Motion Estimation (used for video encoding)
  - Great fit for GPU because these algorithms are very parallel
    - Most operate on blocks of pixels

- Function available in OpenCV library
Towards ADAS: Motion Example
Towards ADAS: Motion Segmentation

- Segment blocks of pixels which have a motion vector with:
  - High confidence
  - High enough magnitude
  - Similar direction

- These segments represent the “foreground”
  - Objects which are moving faster than those around them
Towards ADAS: Geometry Reduction

- Reducing the classification search space using geometric constraints
  - Pedestrians cannot be taller than a certain height
  - Pedestrians cannot be shorter than a certain height
  - Pedestrians cannot be detached from the ground

- What is needed?
  - Estimate of the ground plane
  - Vehicle / camera pitch information
Towards ADAS: Geometry Reduction

Classification Bounding Box

Tracking Bounding Box

Max. allowed height

Estimated Horizon

Min. allowed height

Estimated pedestrian base
Pedestrian tracking

- Why do we need to track?
  - Classification does not give us successful results at each frame
  - Gives us a better approximation of a pedestrian’s trajectory

- Tracking using motion information
  - Median Flow

- Closed-loop tracking
  - MeanShift, CamShift, and TemplateMatching

- Function available in OpenCV library
Tracking: MeanShift

Histogram of tracking box in Hue space
**Traditional Academic Pipeline**

1. Input Frame
2. Preprocessing
3. Classification
4. False Positive Reduction
5. Tracking

<table>
<thead>
<tr>
<th>Stage</th>
<th>HW</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Frame</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Preprocessing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>False Positive Reduction</td>
<td></td>
<td>15x</td>
</tr>
<tr>
<td>Tracking</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Proposed Optimized Pipeline**

1. Input Frame
2. Preprocessing
3. Motion Estimation
4. Geometry Reduction
5. Depth/Motion Segmentation
6. Classification
7. Tracking

<table>
<thead>
<tr>
<th>Stage</th>
<th>HW</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Frame</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Preprocessing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motion Estimation</td>
<td></td>
<td>Free</td>
</tr>
<tr>
<td>Geometry Reduction</td>
<td></td>
<td>8x</td>
</tr>
<tr>
<td>Depth/Motion Segmentation</td>
<td></td>
<td>44x</td>
</tr>
<tr>
<td>Classification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracking</td>
<td></td>
<td>15x</td>
</tr>
</tbody>
</table>
Demo

- Video demo
Integrated ADAS Performance

- **Monocular results**

<table>
<thead>
<tr>
<th>NVIDIA GeForce GTX 470 (14 SMs)</th>
<th>NVIDIA GeForce GT 640 (2 SMs)</th>
<th>Target Automotive GPU (1 SM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 fps</td>
<td>7.14 fps</td>
<td>3.57 fps</td>
</tr>
</tbody>
</table>

- **Stereo stixels + ground plane results**
  - **Motion Estimation & Geometric Reduction**

<table>
<thead>
<tr>
<th>NVIDIA GeForce GTX 470 (14 SMs)</th>
<th>NVIDIA GeForce GT 640 (2 SMs)</th>
<th>Target Automotive GPU (1 SM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>135 fps</td>
<td>19.29 fps</td>
<td>9.64 fps</td>
</tr>
</tbody>
</table>

Benenson et al - “Pedestrian detection at 100 frames per second”
Conclusions

- Full-frame classification is not fast enough yet
- GPU acceleration can lead to a large speed-up in classification
- Classification search space must be significantly reduced to achieve real-time results
- Recent progress in academic research is employing practical system deployment concepts
- Advances in sensors and processors will enable very high frame rates which will free up resources for other tasks
Thank You!

- Special thanks to
  - Vladimir Glavtchev
  - Shalini Gupta
  - Phillip Smith
  - Elif Albuz