Rapid Object Detection using a Boosted Cascade of Simple Features
(Viola and Jones CVPR 2001)

Presented by Muhammad Zubair Malik
25 Jan 2007

Object Detection

- Detection: Given an image where are the objects (faces), if any?
- Faces are rare, 1000 times less than non-faces.
- A good detection scheme keeps false positive rate less than true positive
- Why Detect: user interfaces, interactive agents, security systems, video compression, image database analysis.

Why Face Detection is Difficult?*

- Pose: Variation due to the relative camera-face pose (frontal, 45 degree, profile, upside down), and some facial features such as an eye or the nose may become partially or wholly occluded.
- Presence or absence of structural components: Facial features such as beards, mustaches, and glasses may or may not be present, and there is a great deal of variability amongst these components including shape, color, and size.
- Facial expression: The appearance of faces are directly affected by a person's facial expression.
- Occlusion: Faces may be partially occluded by other objects. In an image with a group of people, some faces may partially occlude other faces.
- Image orientation: Face images directly vary for different rotations about the camera's optical axis.
- Imaging conditions: When the image is formed, factors such as lighting (spectra, source distribution and intensity) and camera characteristics (sensor response, lenses) affect the appearance of a face.

* Taken from Bill Freeman whom I think copied it from ICPR 04 tutorial on detection

Contributions of this paper

- A very fast and robust object detection framework.
- A very simple set of Haar like box features
- A commensurating Image representation (that enables fast calculation of features, feature scaling and normalization)
- Efficient feature selection based on boosting
- Attentional Cascading of classifiers that spends more time on promising regions.

Training Process

- Select feature with least weighted error
- Update the weights with exp loss
- Exit if Detect and FP rates met

Detection

- Selection of a Strong Classifier: The sum of weak classifiers

Image Sub-window

Classifier 1
Classifier 2
Classifier 3

Non-Face
Non-Face
Non-Face
FACE

Normalization and Scaling

Already Selected Features

Normalized Feature

Its Polarity

Its Threshold

Already Selected Features

A Strong Classifier: The sum of weak classifiers

Detection Rate False Positive Rate

Pool of Weak Haar like Features

Weight Initialized Training Process

Faces
Non-Faces

Pool of Weak Haar like Features

- Select feature with least weighted error
- Update the weights with exp loss
- Exit if Detect and FP rates met

* Above and other images in this presentation are taken from: http://people.csail.mit.edu/billf/learningvision/Viola-ICCV-tutorial.ppt
Features

- "Rectangle filters" Similar to Haar wavelets
- Many times over complete
- Encode ad-hoc domain knowledge
- Allow fast operations
- Coarse compared to many filters but rich enough for learning

Integral Image

- Allows fast computation of rectangular features
- At location x, y contains the sum of pixels above and to the left of x, y, inclusive.
- Can be computed in one pass using the recurrence
  \[
  i(x, y) = \sum_{(x', y')} (x', y')
  \]
  \[
  ii(x, y) = ii(x - 1, y) + ii(x, y)
  \]

Classification Function \( h_j(x) \)

- A weak classifier (just better than chance)
- Consists of a single rectangular feature \( f_j \)
- A threshold \( \Theta_j \) for detection
- A polarity \( p_j \) indicating the direction of inequality

\[
\begin{align*}
  h_j(x) & = 1 \text{ if } p_j f_j(x) < p_j \Theta_j \\
  & = 0 \text{ otherwise}
\end{align*}
\]

AdaBoost for Efficient Feature Selection

- Intuition: When combing multiple independent and diverse decisions each of which is at least more accurate than random guessing, random errors cancel each other out, correct decisions are reinforced.
- Examples are given weights. At each iteration, a new hypothesis is learned and the examples are reweighted to focus the system on examples that the most recently learned classifier got wrong.
- Iteratively combine classifiers, weighted by their error.
- Training error converges to 0 quickly
- Test error is related to training margin

AdaBoost Toy Example

* Copied from Freund & Shapire
Feature Selection

• Features = Weak Classifiers
• Each round selects the optimal feature given:
  – Previous selected features
  – Exponential Loss
• In each round for each remaining feature sum of weighted errors is evaluated for all examples
• The classifier with lowest error is selected
• Reweight the examples misclassified by lowest error classifier
• Finally build a weighted sum of classifiers

Example Face Classifier

• A classifier with 200 rectangle features was learned using AdaBoost
• 95% correct detection on test set with 1 in 14084 false positives.
• To directly improve performance we need to add more features to the classifier but that increases computation time.
• The initial features selected even have meaningful interpretation.

The Attentional Cascade

• Increases detection performance while reducing the computation time
• A very simple classifier can be built that rejects many negative windows while keeping all positive ones
• We could define a computational risk hierarchy a nested set of classifier classes
• Initial simple classifiers minimize false positives for later complicated classifiers

Building a Fast Classifier

• Given a nested set of classifier hypothesis classes
• Computational Risk Minimization

The ROC (receiver operating characteristic) Curve

• A 1 feature classifier achieves 100% detection rate and about 50% false positive rate.
• A 5 feature classifier achieves 100% detection rate and 40% false positive rate (20% cumulative)
  – using data from previous stage.
• A 20 feature classifier achieve 100% detection rate with 10% false positive rate (2% cumulative)
Training the Cascade Classifier

• Trade off between number of features and
  i. The number of classifier stages
  ii. The number of features in each stage
  iii. Threshold of each stage
• Actually a target is selected for minimum reduction in false positive and maximum decrease in target detection. We add features till the target is met.

Accuracy of face detector

• Performance on MIT+CMU test set containing 130 images with 507 faces and about 75 million sub-windows.

Comparison with other systems

<table>
<thead>
<tr>
<th></th>
<th>10</th>
<th>31</th>
<th>50</th>
<th>65</th>
<th>78</th>
<th>95</th>
<th>110</th>
<th>167</th>
<th>422</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viola-Jones</td>
<td>78.3</td>
<td>85.2</td>
<td>88.8</td>
<td>90.0</td>
<td>90.1</td>
<td>90.8</td>
<td>91.1</td>
<td>91.8</td>
<td>93.7</td>
</tr>
<tr>
<td>Rowley-Baluja-Kanade</td>
<td>83.2</td>
<td>89.0</td>
<td>91.4</td>
<td>91.8</td>
<td>92.2</td>
<td>90.1</td>
<td>89.9</td>
<td>89.9</td>
<td>94.4</td>
</tr>
<tr>
<td>Schneiderman-Kanade</td>
<td>94.4</td>
<td>94.4</td>
<td>94.4</td>
<td>94.4</td>
<td>94.4</td>
<td>94.4</td>
<td>94.4</td>
<td>94.4</td>
<td>94.4</td>
</tr>
<tr>
<td>Roth-Yang-Ahuja</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Output of Face Detector on Test Images

Feature Localization

• Surprising properties of this framework
  – The cost of detection is not a function of image size
    • Just the number of features
  – Learning automatically focuses attention on key regions
  • The “feature” detector can include a large contextual region around the feature

Speed of Face Detector

• Speed is proportional to the average number of features computed per sub-window.
• On the MIT+CMU test set, an average of 9 features out of a total of 6061 are computed per sub-window.
• On a 700 Mhz Pentium III, a 384x288 pixel image takes about 0.067 seconds to process (15 fps).
• Roughly 15 times faster than Rowley-Baluja-Kanade and 600 times faster than Schneiderman-Kanade.
Conclusions

• Authors had developed the fastest known face detector for gray scale images
• Three contributions with broad applicability
  – Cascaded classifier yields rapid classification
  – AdaBoost as an extremely efficient feature selector
  – Rectangle Features + Integral Image can be used for rapid image analysis