Haar-like Features and Integral Image Representation

Master in Computer Vision and Artificial Intelligence
18th December 2009

David Gerónimo

dgeronimo@cvc.uab.es

www.cvc.uab.es/adas
Index

- Haar features
  - Definition
  - Inspiration
  - Normalization
  - Different sets

- Problem
  - Computational cost
  - Definition

- Integral image representation

- Further tricks and extensions
  - Integral image for 45° rotated features
  - Integral image for almost any angle feature
  - Integral volume
  - Commutativity property

- Applications
Haar features

*Computer Vision in the late 90s...*

- Computationally expensive features: raw pixels, edges, etc.
Haar features

Computer Vision in the late 90s...

- Computationally expensive features: raw pixels, edges, etc.
- Inadequate image representations (features) to deal with complex objects: difficult to model with existing features.
Haar features

IEEE Conference on Computer Vision and Pattern Recognition 1997

Pedestrian Detection Using Wavelet Templates

Michael Oren  Constantine Papageorgiou  Pawan Sinha  
Edgar Osuna  Tomaso Poggio

CBCI and AI Lab  MIT  
Cambridge, MA 02139

Abstract

This paper presents a trainable object detection architecture that is applied to detecting people in static images of cluttered scenes. This problem poses several challenges. People are highly non-rigid objects with a high degree of variability in size, shape, color, and texture. Unlike previous approaches, this system learns from examples and does not rely on any a priori (handcrafted) models or on motion.

The detection technique is based on the novel idea of the wavelet template that defines the shape of an object in terms of a subset of the wavelet coefficients of the image. It is invariant to changes in color and texture and can be used to robustly define a rich and complex class of objects such as people. We show how the invariant properties and computational efficiency of the wavelet template make it an effective tool for object detection.

1 Introduction

The problem of object detection has seen a high degree of interest over the years. The fundamental problem is how to characterize an object class. In contrast to the ease of pattern classification, where we need to decide between a relatively small number of classes, the detection problem requires us to differentiate between the object class and the rest of the world. As a result, the class description for object detection must have large discriminative power to handle the cluttered scenes it will be presented with. Furthermore, in modeling complicated classes of objects (e.g., faces, redrafted models. An important aspect of our system is that the model is automatically learned from examples and avoids the use of motion and explicit segmentation.

One of the successful systems in the area of trainable object detection in cluttered scenes is the face detection system of Sung and Poggio [15]. They model face and non-face patterns in a high dimensional space and derive a statistical model for the class of frontal human faces. Similar face detection systems have been developed by others (Vaillant, et al. [17], Rowley, et al. [11], Moghaddam and A. Pentland [8], Osuna et al. [3]).

Frontal human faces, despite their variability, share very similar patterns (shape and the spatial layout of facial features) and their color space is very constrained. This is not the case with pedestrians. Figure 1 shows several typical images of people in our database. These images illustrate the difficulties of pedestrian detection; there is significant variability in the patterns and colors within the boundaries of the body. The detection problem is also complicated by the absence of constraints on the image background. Given these problems, direct analysis of pixel characteristics (e.g., intensity, color and texture) is not adequate. This paper presents a new approach motivated by an earlier piece of work by one of the authors [12].

[Image of a few different face-regions. This design]
Haar features, also known as Haar-like features, are a simple and inexpensive image features based on intensity differences between rectangle-based regions that share similar shapes to the Haar wavelets.

\[ F_{Haar} = E(R_{\text{white}}) - E(R_{\text{black}}) \]
Haar features, also known as Haar-like features, are a simple and inexpensive image features based on intensity differences between rectangle-based regions that share similar shapes to the Haar wavelets.

$$F_{\text{Haar}} = 12700 - 3400 = 9300$$

[Oren97,Papageorgiou00]
Haar features, also known as Haar-like features, are a simple and inexpensive image features based on intensity differences between rectangle-based regions that share similar shapes to the Haar wavelets.

- High inter-class variability
- Low intra-class variability
- Local oriented intensity differences
- Different scales
- Computationally efficient

(Yale Face Database)
Haar features

Inspired in Haar wavelets (by Dr. Alfréd Haar, 1910)

Wavelet function:
\[ \psi(t) = \begin{cases} 
1 & 0 \leq t < 1/2 \\
-1 & 1/2 \leq t < 1 \\
0 & \text{otherwise}
\end{cases} \]

Scaling function:
\[ \phi(t) = \begin{cases} 
1 & 0 \leq t < 1 \\
0 & \text{otherwise}
\end{cases} \]
Haar features

Normalization (monotonic illumination changes)

\[
F_{Haar} = \frac{E(R_{black}) - E(R_{white})}{\sqrt{E(R_\mu)^2 - E(R_\mu^2)}}
\]

Standard Deviation

\[\text{[Oren97, Papageorgiou00]}\]
Haar features

Normalization (monotonic illumination changes and scale)

\[
F_{Haar} = \frac{E(R_{\text{black}}) - E(R_{\text{white}})}{w \cdot h \cdot \sqrt{E(R_{\mu})^2 - E(R_{\mu}^2)}}
\]

[Oren97, Papageorgiou00]
Haar features

Normalization (absolute value feature)

\[ F_{Haar} = \frac{E(R_{black}) - E(R_{white})}{w \cdot h \cdot \sqrt{E(R^2) - E(R^2)}} \]

Black trousers on gray background

White trousers on gray background

[Oren97, Papageorgiou00]
Haar features

Different sets

Basic Haar set

Haar-like features

Extended Haar features

Borders

Lines

Center-surround

[Papageorgiou00, Viola01, Lienhart02]
The problem

Face detection

"The simplest detection procedure consists in scanning the input image $I$ with $S$ windows and using a face model $M$ (e.g., $K$ Haar features) to classify each window $W$."
The problem

Computational cost for a small region

Equation

\[ \text{sum} = \sum_{x=1}^{10} \sum_{y=1}^{10} I(x, y) \]

Cost

100 memory accesses
99 sums
The problem

Computational cost for a big region

Equation

\[ \sum_{x=1}^{100} \sum_{y=1}^{100} I(x, y) \]

Cost

10,000 memory accesses
9,999 sums
The problem

Computational cost for any region

\[
\text{sum} = \sum_{x=1}^{w} \sum_{y=1}^{h} I(x, y)
\]

Cost

\(O(n)\)

\(w \cdot h\) memory accesses
\(w \cdot h - 1\) sums
The problem

Computational cost for many regions

Model of $N=100$ regions of $50 \times 50$ (w·h) pixels each, $S=500$ windows to scan the input image:

- 125,000,000 accesses
- $\sim 125,000,000$ sums

(per image)

Cost / image

\[
\text{mem accesses} = (w_1 \cdot h_1 + w_2 \cdot h_2 + \cdots + w_n \cdot h_n)
\]

\[
\text{sums} \approx (w_1 \cdot h_1 + w_2 \cdot h_2 + \cdots + w_n \cdot h_n)
\]
The problem

Definition

We want to detect the faces in an image by scanning the input image with a model based on Haar features (N rectangular regions each). How to perform the detection in constant time, that is, O(1), independently of the size and position of the regions? Tip: can we put something between the original image and each rectangular region so the computation of each window gets faster?

you got 5 minutes to think…
**Integral image representation**

**Definition**

*We can use the integral image, also known as summed area table, which allows to compute each region with 4 accesses and 3 sums.*
Integral image representation

Integral image construction

Original image ($I$) Integral image ($ii$)

\[ ii_p = ii_2 + ii_3 - ii_1 + I_p \]

(single pass) [Crow84, Viola01]
Integral image representation

Integral image construction

Original image ($I$) | Integral image ($ii$)
---|---
7 7 7 7 4 8 4 1 | 7 14 21 28 32 40 44 45
7 7 7 8 4 4 1 1 | 14 28 42 57 65 77 82 84
7 7 8 4 8 4 4 1 | 21 42 64 83 99 119 124 127
7 8 4 7 8 4 1 1 | 28 57 83 109 133
7 8 7 7 4 8 1 1 | 133 = 99 + 109 − 83 + 8
7 8 7 7 8 8 1 1

(single pass) [Crow84,Viola01]
Integral image representation

Integral image construction

Original image ($I$)  Integral image ($ii$)

[Crow84, Viola01]
Integral image representation

Integral image retrieval

Original image ($I$)  Integral image ($ii$)

[Crow84, Viola01]
Integral image representation

Integral image retrieval

Original image ($I$)  Integral image ($ii$)

[Crow84, Viola01]
Integral image representation

Integral image retrieval

Integral image \((ii)\)

\[\begin{align*}
ii_1 &= \text{sum}(A) \\
ii_2 &= \text{sum}(A) + \text{sum}(B) \\
ii_3 &= \text{sum}(A) + \text{sum}(C) \\
ii_4 &= \text{sum}(A) + \text{sum}(B) + \text{sum}(C) + \text{sum}(D)
\end{align*}\]

\[
\text{sum}(D) = ii_4 + ii_1 - ii_2 - ii_3
\]

[Crow84, Viola01]
### Integral image representation

#### Integral image retrieval

**Integral image (\(ii\))**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>14</th>
<th>21</th>
<th>28</th>
<th>32</th>
<th>40</th>
<th>44</th>
<th>45</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>28</td>
<td>42</td>
<td>57</td>
<td>65</td>
<td>77</td>
<td>82</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>42</td>
<td>64</td>
<td>83</td>
<td>99</td>
<td>119</td>
<td>124</td>
<td>127</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>57</td>
<td>83</td>
<td>109</td>
<td>133</td>
<td>153</td>
<td>159</td>
<td>163</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>72</td>
<td>105</td>
<td>138</td>
<td>166</td>
<td>194</td>
<td>201</td>
<td>206</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>87</td>
<td>127</td>
<td>167</td>
<td>203</td>
<td>239</td>
<td>247</td>
<td>253</td>
<td></td>
</tr>
</tbody>
</table>

**Original image (\(I\))**

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>4</td>
<td>8</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>8</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[54 = 194 + 42 - 77 - 105\]

[Crow84, Viola01]
Integral image representation

Computational cost for many regions

Example

Model of $N=100$ regions of $50 \times 50$ (w·h) pixels each, $S=500$ windows to scan the $200 \times 200$ input image:

- $360,000$ accesses
- $\sim 360,000$ sums

(10$^5$ vs 10$^8$)

(per image)

(This is obviously a simplification, we should take into account the number of rectangles inside each feature)

Cost / image

$$\text{mem accesses} = 4I_wI_h + S(4N)$$

$$\text{sums} \approx 3I_wI_h + S(3N)$$

[Crow84, Viola01]
Further tricks and extensions

Integral Image for 45° rotated features

Integral image
(summed area table)

[Lienhart02]
Further tricks and extensions

Integral Image for 45° rotated features

Integral image
(summed area table)

[Lienhart02]
Further tricks and extensions

Integral Image for 45° rotated features

Integral image (summed area table)

Rotated integral image (rotated summed area table)

[Lienhart02]
Further tricks and extensions

Integral Image for 45° rotated features

Rotated integral Image construction

Original image ($I$)

Rotated integral image ($rii$)

$rii_p = rii_2 + rii_3 - rii_1 + I_p$

(1st pass)

[18]
Further tricks and extensions

Integral Image for 45° rotated features

Rotated integral Image construction

Original image ($I$)

Rotated integral image ($rii$)

$$rii_p = rii_p + rii_1 - rii_2$$

(2nd pass)

[Lienhart02]
Integral image representation

Integral image for 45° rotated features

Rotated integral Image retrieval

Rotated integral image ($r_{ii}$)

$r_{ii_1} = \text{sum}(A)$

$r_{ii_2} = \text{sum}(A) + \text{sum}(B)$

$r_{ii_3} = \text{sum}(A) + \text{sum}(C)$

$r_{ii_4} = \text{sum}(A) + \text{sum}(B) + \text{sum}(C) + \text{sum}(D)$

$$\text{sum}(D) = r_{ii_p} + r_{ii_1} - r_{ii_2} - r_{ii_3}$$

[Lienhart02]
Further tricks and extensions

Integral image for *almost any* angle feature

0° 45° 11.3° 26.5° 45° 63.4° 78.6°
Further tricks and extensions

Integral image for *almost any* angle feature

Unit-integer rotated integral image

- Just N-1 or 1-N aspects
- 1 rii per angle
- Implementation issues (precision)

![Diagram showing rotated integral images with angles and aspect ratios.](image-url)

[Image: Image showing rotated integral images with angles and aspect ratios.

---

[Messom09]
Further tricks and extensions

Integral image for *almost any* angle feature

Unit-integer rotated integral image

- Just N-1 or 1-N aspects
- 1 rii per angle
- Implementation issues (precision)
Further tricks and extensions

Integral Histograms

Difference in orientation between \ and – angles in the region?

(Edge Orientation Histograms)

[Levi04]
Further tricks and extensions

Integral volume

[Ke05]
Further tricks and extensions

Pixel-wise operations

Original image ➔ Pixel-wise Image Processing ➔ Transformed Image

(Pixel Interpolation, gaussian smooth, whatever..)

[Wang09]
Further tricks and extensions

Pixel-wise operations

Original image → Pixel-wise Image Processing → Transformed Image

(Pixel Interpolation, gaussian smooth, whatever..)

Original image → Pixel-wise Image Processing → Integral Image Construct → Integral Image Retrieval → Transformed Image

(Pixel Interpolation, gaussian smooth, whatever..)

[Wang09]
Applications

Face detection

Eye detection

[Wang05]
Applications

Vehicle detection

Pedestrian detection

[Gerónimo07]

[Gerónimo07]
Conclusions

- Haar features provide simple and efficient image representation to perform detection, recognition, classification, etc.

- The features require normalization to be robust.

- Integral image is useful to fastly compute the summed regions used in Haar features.

- There are extensions of the basic set that can provide improved performance to the classification (for example, extended sets, rotated features, integral volume, etc.)
References

[F. Crow. Summed area tables for texture mapping, SIGGRAPH, pp. 207—212, 1984.]


[K. Levi and Y. Weiss. Learning object detection from a small number of examples: the importance of good features. CVPR04]

[R. Lienhart and J. Maydt. Aan extended set of Haar-like features for rapid object detection, CVPR, 2002.]


[D. Ponsa and A.M. López. A cascade of classifiers for vehicle detection, ACIVS, 2007.]


[P. Viola and M. Jones. Rapid object detection using a cascade of simple features, CVPR, 2001.]


Haar-like Features and Integral Image Representation

Master in Computer Vision and Artificial Intelligence
18th December 2009

David Gerónimo

dgeronimo@cvc.uab.es
www.cvc.uab.es/adas