Fast object detection techniques backed with integral images

Przemysław Klęsk

Faculty of Computer Science and Information Technology
West Pomeranian University of Technology
ul. Żołnierska 49, 71-210 Szczecin, Poland
pklesk@wi.zut.edu.pl

Acknowledgement

- Research project financed by the National Science Centre (2017–2020, no.: 2016/21/B/ST6/01495).
- Title: *"Constructions of sets of integral images for fast feature extraction and machine learning in detection tasks"*.
- Research team: P. Klęsk, M. Korzeń, A. Bera, D. Sychel, J. Klimaszewski.

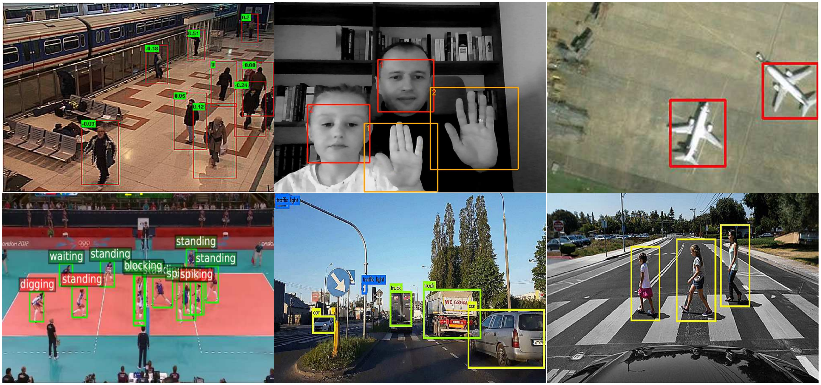
Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

Object detection — applications



faces, people, pedestrians, hands, eyes, vehicles, road signs, traffic lights, licence plates, airplanes, airfield objects, landmines, sport objects, . . .

Feature extraction approaches

Features representing physical quantities (high-level, domain-specific, "manually designed")

- E.g. for face recognition: eyes spread, nose length, forehead–mouth distance, fringe, glasses, gender, age, ethnicity, etc.
- Sophisticated, refined features (computationally expensive).
- May require many low-level techniques (edges, corners, blobs, color segmentation, etc.).
- Fairly small feature sets for machine learning ($\sim 10^2$).
- Applicable in *recognition* tasks (suitable image fragment given as input, numerous classes).
- *Not* applicable in *detection* tasks (whole image as input, 2 classes: "target vs. non-target").

Features representing simple geometric properties (low-level, "automatic", "learned")

- Examples: raw pixels + PCA, LBP, Haar-like features, texture coding, bag of words, HOG, moments (geometric, statistical, Fourier, Fourier–Mellin), neural networks (CNNs), etc.
- Oriented towards simple description of shape (computationally cheap . . . mostly).
- Large feature sets at learning stage ($\sim 10^4, 10^5$) — "brute-force attack".
- Connection between simple features and classes might be unclear for designer.
- Learning algorithm expected to select a subset of relevant features ($\sim 10^3$).
- Applicable in *recognition* tasks. Only *some* applicable directly in *detection* tasks.

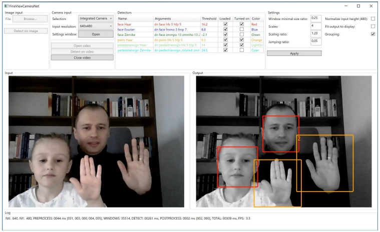
Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Object detection — overview

- **Dense detection** procedures have **high computational demands** (e.g. ≈ $10^{5\pm1}$ **image windows to be analyzed** per frame, under 1 s or less)
- **Procedure** (sketch):

```
loop over several scales;
    for each scale scan image with a sliding window;
        for each window position: (1) extract features, (2) calculate classifier's response.
```

- Common repertoire: **Haar-like** features or **HOG** descriptor.
- Features of windows extracted fast, in **constant time** — ***O*(1)** — owing to a computational trick known as: **integral image**.
- Number of operations does *not* depend on the number of pixels in given window.
- Haar/HOG drawbacks: **sometimes not accurate enough**, **rotationally dependent**.

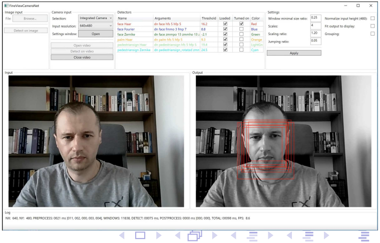


Selection: Integrated Camera
Input resolution: 640x480
Settings window: Open

Name	Arguments	Threshold	Loaded	Turned on	Color
face Haar	dn face hfs 5 hfp 5	16.2	☑	☑	Red
face Fourier	dn face fmmo 3 fmp 7	8.8	☑	☐	Blue
face Zernike	dn face zmmpo 13 zmmho 13 z	-2.1	☑	☐	Green
palm Haar	dn palm hfs 5 hfp 5	9.3	☑	☑	Orange
pedestriansign Haar	dn pedestriansign hfs 5 hfp 5	14	☑	☑	LightGr
pedestriansign Zernike	dn pedestriansign_rotated zmn	24.5	☑	☐	Cyan

Window minimal size ratio: 0.25
Scales: 4
Scaling ratio: 1.20
Jumping ratio: 0.05
Normalize input height (480): ☐
Fit output to display: ☐
Grouping: ☑

```
NX:  640, NY:  480, PREPROCESS: 0044 ms [031, 003, 000, 004, 005], WINDOWS: 35514, DETECT: 00261 ms, POSTPROCESS: 0002 ms [002, 000], TOTAL: 00309 ms, FPS:  3.3
```



Selection: Integrated Camera
Input resolution: 640x480
Settings window: Open

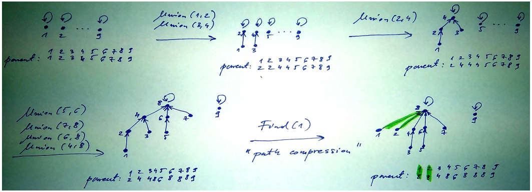
Name	Arguments	Threshold	Loaded	Turned on	Color
face Haar	dn face hfs 5 hfp 5	16.2	☑	☑	Red
face Fourier	dn face fmmo 3 fmp 7	8.8	☑	☐	Blue
face Zernike	dn face zmmpo 13 zmmho 13 z	-2.1	☑	☐	Green
palm Haar	dn palm hfs 5 hfp 5	9.3	☑	☐	Orange
pedestriansign Haar	dn pedestriansign hfs 5 hfp 5	19.4	☑	☐	LightGr
pedestriansign Zernike	dn pedestriansign_rotated zmn	24.5	☑	☐	Cyan

Window minimal size ratio: 0.25
Scales: 4
Scaling ratio: 1.20
Jumping ratio: 0.05
Normalize input height (480): ☐
Fit output to display: ☐
Grouping: ☐

```
NX:  640, NY:  480, PREPROCESS: 0021 ms [011, 002, 000, 003, 004], WINDOWS: 11838, DETECT: 00075 ms, POSTPROCESS: 0000 ms [000, 000], TOTAL: 00098 ms, FPS:  8.6
```

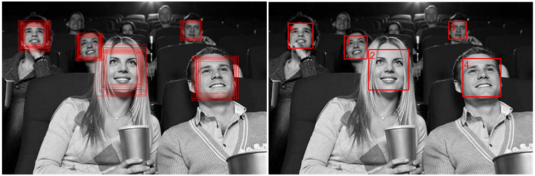
Constant-time computations

- $O(1)$ — the most attractive for a computer scientist, but rare . . .
- Typically: amortized constant-time complexity.
- hash tables, hash maps
- Union–Find data structure



— amortized 'Find' operation takes $O(\log^* n)$ — *iterated logarithm*, note: $\log_2^* 2^{65\,536} = 5$

- Haar-like features (Viola & Jones, 2001, 2004), HOG descriptor (Dalal & Triggs, 2005).



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Cost of detection procedure

- **Cost (direct):**

$$\Theta\Bigg(\sum_{s=0}^{S-1}\underbrace{\frac{n_x-\alpha^s w_x+1}{\beta\alpha^s w_x}}_{dx(s)}\underbrace{\frac{n_y-\alpha^s w_y+1}{\beta\alpha^s w_y}}_{dy(s)}\Big(n\,\underbrace{\alpha^s w_x\alpha^s w_y}_{\text{no. of pixels}}c_{fe/px}+n\,c_{d/f}\Big)\Bigg), \qquad (1)$$

  $n_x \times n_y$ — image dimensions,
  $w_x \times w_y$ — smallest size of sliding window,
  S — number of scales, α — window growing factor, β — window jumping ratio,
  $d_x(s)$, $d_y(s)$ — window jumps for given scale,
  n — number of selected features,
  $c_{fe/px}$ — average cost of 1 feature extraction per pixel,
  $c_{d/f}$ — average cost of classifier's response per 1 feature.

- **Example:**
  $n_x = 640$, $n_y = 480$,
  $w_x = w_y = 48$, (smallest objects ≈ 10% of image height)
  $S = 8$, $\alpha = 1.2$, $\beta = 0.05$,
  $n = 500$ (suppose it is sufficient for our targets)
  $c_{fe/px} = c_{d/f} = 10^{-10}$ s, (optimistic)
  → **windows to be analyzed ≈ 126 000** (neglecting roundings of window sizes and jumps)
  → **time ≈ 33 s**.

Ideas for improvements

1 Integral images (cumulants)

- One or more integral images prepared once before detection procedure (or possibly, before each scale scan — if scale-dependent).
- Features extracted in constant time c_{fe}, regardless of number of pixels in window.
- Complexity reduced to:

$$\Theta\left(\sum_{s=0}^{S-1} \frac{n_x - \alpha^s w_x + 1}{\beta\alpha^s w_x} \frac{n_y - \alpha^s w_y + 1}{\beta\alpha^s w_y} \left(n\, c_{fe} + n\, c_{d/f}\right)\right). \quad (2)$$

2 Classifiers cascade

- Observation: positive windows constitute a very small fraction of all windows,
- Classifier "split" into stages (layers), applying succesively more features.
- Positive indication requires traversing all stages. Negative indication on any stage stops further analysis.
- Average number $\overline{n}$ of features per window much smaller than the total: $(\bar{n} \ll n)$.
- Complexity reduced to:

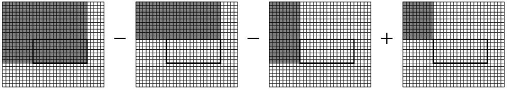
$$\Theta\left(\sum_{s=0}^{S-1} \frac{n_x - \alpha^s w_x + 1}{\beta\alpha^s w_x} \frac{n_y - \alpha^s w_y + 1}{\beta\alpha^s w_y} \left(\overline{n}\, c_{fe} + \overline{n}\, c_{d/f}\right)\right). \quad (3)$$

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Integral images

Known examples:

- $ii(x,y) = \sum_{1\leqslant j\leqslant x} \sum_{1\leqslant k\leqslant y} i(j,k)$ → **fast (constant-time) sums or averages** of pixels,



$$\sum_{x_1\leqslant x\leqslant x_2} \sum_{y_1\leqslant y\leqslant y_2} i(x,y) = ii(x_2,y_2) - ii(x_1-1,y_2) - ii(x_2,y_1-1) + ii(x_1-1,y_1-1)$$

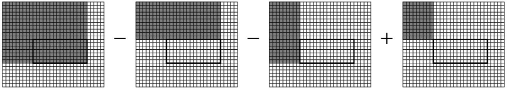
- $ii(x,y) = \sum_{1\leqslant j\leqslant x} \sum_{1\leqslant k\leqslant y} i^2(j,k)$ → **fast (constant-time) variances** of pixels,
- $ii_l(x,y) = \sum_{1\leqslant j\leqslant x} \sum_{1\leqslant k\leqslant y} v(j,k,l)$ → **fast (constant-time) angle votes** in sections l=$1,2,\ldots$ (HOG).

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Integral images

Known examples:

- $ii(x,y) = \sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} i(j,k)$ → **fast (constant-time) sums or averages** of pixels,



$$\sum_{x_1\leqslant x\leqslant x_2}\sum_{y_1\leqslant y\leqslant y_2} i(x,y) = ii(x_2,y_2) - ii(x_1-1,y_2) - ii(x_2,y_1-1) + ii(x_1-1,y_1-1)$$

- $ii(x,y) = \sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} i^2(j,k)$ → **fast (constant-time) variances** of pixels,
- $ii_l(x,y) = \sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} v(j,k,l)$ → **fast (constant-time) angle votes** in sections l=$1,2,\ldots$ (HOG).

Our contributions — new repertoire of integral images and features:

- **fast (constant-time) Fourier moments** (Klęsk, 2017),
- **fast (constant-time) statistical moments** (Klęsk & Bera, 2018),
- **fast (constant-time) Zernike moments** (Bera, Klęsk, & Sychel, 2018).

Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

Inner product, norm, orthogonality

- **Inner product** (for functions of one variable):

$$\langle g, h\rangle = \int_0^1 g(x)h(x)\,dx. \tag{4}$$

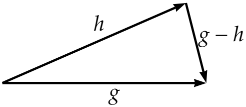
- **Norm** (quadratic) induced by inner product:

$$\|g\| = \sqrt{\langle g, g\rangle} = \left(\int_0^1 g^2(x)\,dx\right)^{1/2}. \tag{5}$$

- **Orthogonality** — g and h are orthogonal, $g \perp h$, iff:

$$\langle g, h\rangle = 0. \tag{6}$$

- Analogies to: vectors, Pythagorean theorem, law of cosines —



$$\|g-h\|^2 = \langle g-h, g-h\rangle = \|g\|^2 - 2\langle g, h\rangle + \|h\|^2. \tag{7}$$

Expansions

- Let f be a function over $[0,1]$.
- Let $g_0, g_1, g_2, \ldots$ be orthogonal functions over $[0,1]$.
- Suppose true is the following representation (expansion):

$$f(x) = c_0 g_0(x) + c_1 g_1(x) + c_2 g_2(x) + \cdots, \tag{8}$$

  where c_k are some real coefficients.
- *What has to happen?* c_k *=?*

Expansions

- Let f be a function over $[0,1]$.
- Let $g_0, g_1, g_2, \ldots$ be orthogonal functions over $[0,1]$.
- Suppose true is the following representation (expansion):

$$f(x) = c_0 g_0(x) + c_1 g_1(x) + c_2 g_2(x) + \cdots, \tag{8}$$

where c_k are some real coefficients.

- *What has to happen?* $c_k = ?$

$$f = c_0 g_0 + c_1 g_1 + c_2 g_2 + \cdots \tag{9}$$

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Expansions

- Let f be a function over $[0, 1]$.
- Let $g_0, g_1, g_2, \ldots$ be orthogonal functions over $[0, 1]$.
- Suppose true is the following representation (expansion):

$$f(x) = c_0 g_0(x) + c_1 g_1(x) + c_2 g_2(x) + \cdots, \tag{8}$$

where c_k are some real coefficients.

- *What has to happen?* c_k *=?*

$$\begin{aligned} f &= c_0 g_0 + c_1 g_1 + c_2 g_2 + \cdots \\ c_k g_k &= f - \sum_{\substack{j=0 \\ j \neq k}}^{\infty} c_j g_j \qquad \textit{(isolating expression with } c_k\textit{)} \end{aligned} \tag{9}$$

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Expansions

- Let f be a function over $[0,1]$.
- Let $g_0, g_1, g_2, \ldots$ be orthogonal functions over $[0,1]$.
- Suppose true is the following representation (expansion):

$$f(x) = c_0 g_0(x) + c_1 g_1(x) + c_2 g_2(x) + \cdots\,, \tag{8}$$

where c_k are some real coefficients.

- *What has to happen?* c_k *=?*

$$\begin{aligned} f &= c_0 g_0 + c_1 g_1 + c_2 g_2 + \cdots \\ c_k g_k &= f - \sum_{\substack{j=0\\ j\neq k}}^{\infty} c_j g_j && \textit{(isolating expression with } c_k\textit{)} \\ c_k \langle g_k, g_k\rangle &= \langle f, g_k\rangle - \sum_{\substack{j=0\\ j\neq k}}^{\infty} c_j \underbrace{\langle g_j, g_k\rangle}_{0} && \textit{(taking inner product } \langle\cdot, g_k\rangle \textit{ sidewise)} \end{aligned} \tag{9}$$

Expansions

- Let f be a function over $[0,1]$.
- Let $g_0, g_1, g_2, \ldots$ be orthogonal functions over $[0,1]$.
- Suppose true is the following representation (expansion):

$$f(x) = c_0 g_0(x) + c_1 g_1(x) + c_2 g_2(x) + \cdots, \tag{8}$$

where c_k are some real coefficients.

- *What has to happen?* $c_k = ?$

$$
\begin{aligned}
f &= c_0 g_0 + c_1 g_1 + c_2 g_2 + \cdots \\
c_k g_k &= f - \sum_{\substack{j=0 \\ j\neq k}}^{\infty} c_j g_j && \textit{(isolating expression with } c_k\textit{)} \\
c_k \langle g_k, g_k \rangle &= \langle f, g_k \rangle - \sum_{\substack{j=0 \\ j\neq k}}^{\infty} c_j \underbrace{\langle g_j, g_k \rangle}_{0} && \textit{(taking inner product } \langle \cdot, g_k \rangle \textit{ sidewise)} \\
c_k &= \frac{1}{\|g_k\|^2} \langle f, g_k \rangle && (9)
\end{aligned}
$$

Approximations in quadratic norm

Theorem 1 ("about the best approximation in quadratic norm")

Let f be a function to be approximated and let $g_0, g_1, \ldots, g_n$ form an orthogonal base. Suppose $\widehat{f} = c_0 g_0 + c_1 g_1 + \cdots + c_n g_n$ is an approximation of f. Then, $\|f - \widehat{f}\|$ is minimum if and only if:

$$c_k = \frac{1}{\|g_k\|^2} \langle f, g_k \rangle. \tag{10}$$

Approximations in quadratic norm

Proof 1 (by necessary condition of optimum):

$$\frac{\partial}{\partial c_k}\|f - \widehat{f}\| = 0$$

Approximations in quadratic norm

Proof 1 (by necessary condition of optimum):

$$\frac{\partial}{\partial c_k}\|f-\widehat{f}\| = 0$$

$$\frac{\partial}{\partial c_k}\left\|f-\sum_{j=0}^{n} c_j g_j\right\|^2 = 0$$

(squared norm can be observed instead of norm)

Approximations in quadratic norm

Proof 1 (by necessary condition of optimum):

$$\frac{\partial}{\partial c_k}\|f-\widehat{f}\| = 0$$

$$\frac{\partial}{\partial c_k}\left\|f-\sum_{j=0}^{n} c_j g_j\right\|^2 = 0$$

(squared norm can be observed instead of norm)

$$\frac{\partial}{\partial c_k}\left\langle f-\sum_{j=0}^{n} c_j g_j, f-\sum_{j=0}^{n} c_j g_j\right\rangle = 0$$

Approximations in quadratic norm

Proof 1 (by necessary condition of optimum):

$$\frac{\partial}{\partial c_k}\|f-\widehat{f}\| = 0$$

$$\frac{\partial}{\partial c_k}\left\|f-\sum_{j=0}^{n}c_j g_j\right\|^2 = 0$$

(squared norm can be observed instead of norm)

$$\frac{\partial}{\partial c_k}\left\langle f-\sum_{j=0}^{n}c_j g_j, f-\sum_{j=0}^{n}c_j g_j\right\rangle = 0$$

$$\frac{\partial}{\partial c_k}\left(\|f\|^2-2\sum_{j=0}^{n}c_j\langle f,g_j\rangle+\sum_{j=0}^{n}\sum_{l=0}^{n}c_j c_l\langle g_j,g_l\rangle\right) = 0$$

Approximations in quadratic norm

Proof 1 (by necessary condition of optimum):

$$\frac{\partial}{\partial c_k}\|f-\widehat{f}\| = 0$$

$$\frac{\partial}{\partial c_k}\Big\|f-\sum_{j=0}^{n} c_j g_j\Big\|^2 = 0$$

(squared norm can be observed instead of norm)

$$\frac{\partial}{\partial c_k}\Big\langle f-\sum_{j=0}^{n} c_j g_j, f-\sum_{j=0}^{n} c_j g_j\Big\rangle = 0$$

$$\frac{\partial}{\partial c_k}\left(\|f\|^2 - 2\sum_{j=0}^{n} c_j\langle f, g_j\rangle + \sum_{j=0}^{n}\sum_{l=0}^{n} c_j c_l\langle g_j, g_l\rangle\right) = 0$$

$$\frac{\partial}{\partial c_k}\left(-2\sum_{j=0}^{n} c_j\langle f, g_j\rangle + \sum_{j=0}^{n} c_j^2\|g_j\|^2\right) = 0$$

(due to orthogonality $\langle g_j, g_l\rangle = 0$ for $j \neq l$)

Approximations in quadratic norm

Proof 1 (by necessary condition of optimum):

$$\frac{\partial}{\partial c_k}\|f-\widehat{f}\|=0$$

$$\frac{\partial}{\partial c_k}\Big\|f-\sum_{j=0}^{n}c_j g_j\Big\|^2=0 \qquad \textit{(squared norm can be observed instead of norm)}$$

$$\frac{\partial}{\partial c_k}\Big\langle f-\sum_{j=0}^{n}c_j g_j, f-\sum_{j=0}^{n}c_j g_j\Big\rangle=0$$

$$\frac{\partial}{\partial c_k}\left(\|f\|^2-2\sum_{j=0}^{n}c_j\langle f,g_j\rangle+\sum_{j=0}^{n}\sum_{l=0}^{n}c_j c_l\langle g_j,g_l\rangle\right)=0$$

$$\frac{\partial}{\partial c_k}\left(-2\sum_{j=0}^{n}c_j\langle f,g_j\rangle+\sum_{j=0}^{n}c_j^2\|g_j\|^2\right)=0 \qquad \textit{(due to orthogonality } \langle g_j,g_l\rangle=0 \textit{ for } j\neq l\textit{)}$$

$$-2\langle f,g_k\rangle+2c_k\|g_k\|^2=0$$

■

Approximations in quadratic norm

Proof 2 (by contradiction): Suppose there exist a better sequence of coefficients $d_0, d_1, \ldots, d_n$ such that for $h = d_0 g_0 + d_1 g_1 + \cdots + d_n g_n$ we have: $\|f - h\| \leqslant \|f - \widehat{f}\|$. Then:

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Approximations in quadratic norm

Proof 2 (by contradiction): Suppose there exist a better sequence of coefficients $d_0, d_1, \ldots, d_n$ such that for $h = d_0 g_0 + d_1 g_1 + \cdots + d_n g_n$ we have: $\|f - h\| \leqslant \|f - \widehat{f}\|$. Then:

$$\|f - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(squared norms observed)}$$

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Approximations in quadratic norm

Proof 2 (by contradiction): Suppose there exist a better sequence of coefficients $d_0, d_1, \ldots, d_n$ such that for $h = d_0 g_0 + d_1 g_1 + \cdots + d_n g_n$ we have: $\|f - h\| \leqslant \|f - \widehat{f}\|$. Then:

$$\begin{aligned} \|f - h\|^2 &\leqslant \|f - \widehat{f}\|^2 && \textit{(squared norms observed)} \\ \|f - \widehat{f} + \widehat{f} - h\|^2 &\leqslant \|f - \widehat{f}\|^2 && \textit{(adding a suitable zero)} \end{aligned}$$

Approximations in quadratic norm

Proof 2 (by contradiction): Suppose there exist a better sequence of coefficients $d_0, d_1, \ldots, d_n$ such that for $h = d_0 g_0 + d_1 g_1 + \cdots + d_n g_n$ we have: $\|f - h\| \leqslant \|f - \widehat{f}\|$. Then:

$$\|f - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(squared norms observed)}$$

$$\|f - \widehat{f} + \widehat{f} - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(adding a suitable zero)}$$

$$\cancel{\|f - \widehat{f}\|^2} + 2\langle f - \widehat{f}, \widehat{f} - h\rangle + \|\widehat{f} - h\|^2 \leqslant \cancel{\|f - \widehat{f}\|^2}$$

Approximations in quadratic norm

Proof 2 (by contradiction): Suppose there exist a better sequence of coefficients $d_0, d_1, \ldots, d_n$ such that for $h = d_0 g_0 + d_1 g_1 + \cdots + d_n g_n$ we have: $\|f - h\| \leqslant \|f - \widehat{f}\|$. Then:

$$\|f - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(squared norms observed)}$$

$$\|f - \widehat{f} + \widehat{f} - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(adding a suitable zero)}$$

$$\cancel{\|f - \widehat{f}\|^2} + 2\langle f - \widehat{f}, \widehat{f} - h\rangle + \|\widehat{f} - h\|^2 \leqslant \cancel{\|f - \widehat{f}\|^2}$$

It suffices to show that the error $f - \widehat{f}$ and the difference of approximators $\widehat{f} - h$ are orthogonal:

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Approximations in quadratic norm

Proof 2 (by contradiction): Suppose there exist a better sequence of coefficients $d_0, d_1, \ldots, d_n$ such that for $h = d_0 g_0 + d_1 g_1 + \cdots + d_n g_n$ we have: $\|f - h\| \leqslant \|f - \widehat{f}\|$. Then:

$$\|f - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(squared norms observed)}$$

$$\|f - \widehat{f} + \widehat{f} - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(adding a suitable zero)}$$

$$\cancel{\|f - \widehat{f}\|^2} + 2\langle f - \widehat{f}, \widehat{f} - h\rangle + \|\widehat{f} - h\|^2 \leqslant \cancel{\|f - \widehat{f}\|^2}$$

It suffices to show that the error $f - \widehat{f}$ and the difference of approximators $\widehat{f} - h$ are orthogonal:

$$\langle f, \widehat{f}\rangle - \langle f, h\rangle - \|\widehat{f}\|^2 + \langle \widehat{f}, h\rangle = \Big\langle f, \sum_{k=0}^{n} c_k g_k \Big\rangle - \Big\langle f, \sum_{k=0}^{n} d_k g_k \Big\rangle - \Big\langle \sum_{k=0}^{n} c_k g_k, \sum_{k=0}^{n} c_k g_k \Big\rangle + \Big\langle \sum_{k=0}^{n} c_k g_k, \sum_{k=0}^{n} d_k g_k \Big\rangle$$

$$= \sum_{k=0}^{n} c_k \underbrace{\langle f, g_k\rangle}_{c_k\|g_k\|^2} - \sum_{k=0}^{n} d_k \underbrace{\langle f, g_k\rangle}_{c_k\|g_k\|^2} - \sum_{k=0}^{n} c_k^2 \|g_k\|^2 + \sum_{k=0}^{n} c_k d_k \|g_k\|^2 = 0.$$

Approximations in quadratic norm

Proof 2 (by contradiction): Suppose there exist a better sequence of coefficients $d_0, d_1, \ldots, d_n$ such that for $h = d_0 g_0 + d_1 g_1 + \cdots + d_n g_n$ we have: $\|f - h\| \leqslant \|f - \widehat{f}\|$. Then:

$$\|f - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(squared norms observed)}$$

$$\|f - \widehat{f} + \widehat{f} - h\|^2 \leqslant \|f - \widehat{f}\|^2 \qquad \textit{(adding a suitable zero)}$$

$$\cancel{\|f - \widehat{f}\|^2} + 2\langle f - \widehat{f}, \widehat{f} - h\rangle + \|\widehat{f} - h\|^2 \leqslant \cancel{\|f - \widehat{f}\|^2}$$

It suffices to show that the error $f - \widehat{f}$ and the difference of approximators $\widehat{f} - h$ are orthogonal:

$$\langle f, \widehat{f}\rangle - \langle f, h\rangle - \|\widehat{f}\|^2 + \langle \widehat{f}, h\rangle = \Big\langle f, \sum_{k=0}^{n} c_k g_k \Big\rangle - \Big\langle f, \sum_{k=0}^{n} d_k g_k \Big\rangle - \Big\langle \sum_{k=0}^{n} c_k g_k, \sum_{k=0}^{n} c_k g_k \Big\rangle + \Big\langle \sum_{k=0}^{n} c_k g_k, \sum_{k=0}^{n} d_k g_k \Big\rangle$$

$$= \sum_{k=0}^{n} c_k \underbrace{\langle f, g_k\rangle}_{c_k \|g_k\|^2} - \sum_{k=0}^{n} d_k \underbrace{\langle f, g_k\rangle}_{c_k \|g_k\|^2} - \sum_{k=0}^{n} c_k^2 \|g_k\|^2 + \sum_{k=0}^{n} c_k d_k \|g_k\|^2 = 0.$$

Therefore, $\|\widehat{f} - h\|^2 \leqslant 0$, which is not true unless $\widehat{f} = h$, hence: $d_0 = c_0, d_1 = c_1, \ldots, d_n = c_n$. ■

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

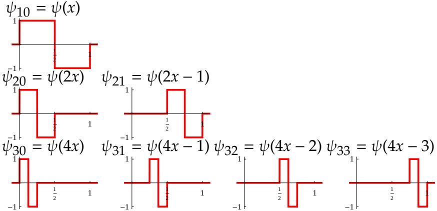
Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

Haar wavelets

- **Mother wavelet and descendants**:

$$\psi(x) = \begin{cases} 1, & 0 \leqslant x < 1/2; \\ -1, & 1/2 \leqslant x < 1; \\ 0, & \text{otherwise.} \end{cases}$$



$$\psi_{j,k} = \psi(2^{j-1}x - k), \qquad j = 1, 2, \ldots; \; k = 0, 1, \ldots, 2^{j-1} - 1.$$

- **Orthogonality**:

$$\forall (j,k) \neq (l,m) \quad \langle \psi_{j,k}, \psi_{l,m} \rangle = \int_0^1 \psi_{j,k}(x)\psi_{l,m}(x)\,dx = 0. \tag{11}$$

- **Expansion** of a function:

$$f(x) = c_0 \cdot 1 + \sum_{j=1}^{\infty} \sum_{k=0}^{2^{j-1}-1} c_{j,k}\psi_{j,k}(x). \tag{12}$$

- **Coefficients**:

$$c_{j,k} = 1/\|\psi_{j,k}\|^2 \langle f, \psi_{j,k} \rangle, \qquad c_0 = \langle f, 1 \rangle. \tag{13}$$

Approximations by Haar wavelets

- Example 1: $f(x) = x$.
- Coefficients:

$$c_0 = \frac{1}{||1||^2}\langle f, 1\rangle = \frac{1}{1}\int_0^1 x\,dx = \frac{x^2}{2}\Big|_0^1 = \frac{1}{2}$$

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Approximations by Haar wavelets

- Example 1: $f(x) = x$.
- Coefficients:

$$c_0 = \frac{1}{||1||^2}\langle f, 1\rangle = \frac{1}{1}\int_0^1 x\,dx = \frac{x^2}{2}\Big|_0^1 = \frac{1}{2}$$

$$c_{1,0} = \frac{1}{||\psi_{1,0}||^2}\langle f, \psi_{1,0}\rangle = \frac{1}{1}\left(\int_{0/2}^{1/2} x\,dx + \int_{1/2}^{2/2}(-x)\,dx\right) = \frac{x^2}{2}\Big|_{0/2}^{1/2} - \frac{x^2}{2}\Big|_{1/2}^{2/2} = -\frac{1}{4}$$

Approximations by Haar wavelets

- Example 1: $f(x) = x$.
- Coefficients:

$$c_0 = \frac{1}{\|1\|^2}\langle f, 1\rangle = \frac{1}{1}\int_0^1 x\,dx = \frac{x^2}{2}\Big|_0^1 = \frac{1}{2}$$

$$c_{1,0} = \frac{1}{\|\psi_{1,0}\|^2}\langle f, \psi_{1,0}\rangle = \frac{1}{1}\left(\int_{0/2}^{1/2} x\,dx + \int_{1/2}^{2/2}(-x)\,dx\right) = \frac{x^2}{2}\Big|_{0/2}^{1/2} - \frac{x^2}{2}\Big|_{1/2}^{2/2} = -\frac{1}{4}$$

$$c_{2,0} = \frac{1}{\|\psi_{2,0}\|^2}\langle f, \psi_{2,0}\rangle = \frac{1}{1/2}\left(\int_{0/4}^{1/4} x\,dx + \int_{1/4}^{2/4}(-x)\,dx\right) = \frac{1}{1/2}\left(\frac{x^2}{2}\Big|_{0/4}^{1/4} - \frac{x^2}{2}\Big|_{1/4}^{2/4}\right) = -\frac{1}{8}$$

$$c_{2,1} = \frac{1}{\|\psi_{2,1}\|^2}\langle f, \psi_{2,1}\rangle = \frac{1}{1/2}\left(\int_{2/4}^{3/4} x\,dx + \int_{3/4}^{4/4}(-x)\,dx\right) = \frac{1}{1/2}\left(\frac{x^2}{2}\Big|_{2/4}^{3/4} - \frac{x^2}{2}\Big|_{3/4}^{4/4}\right) = -\frac{1}{8}$$

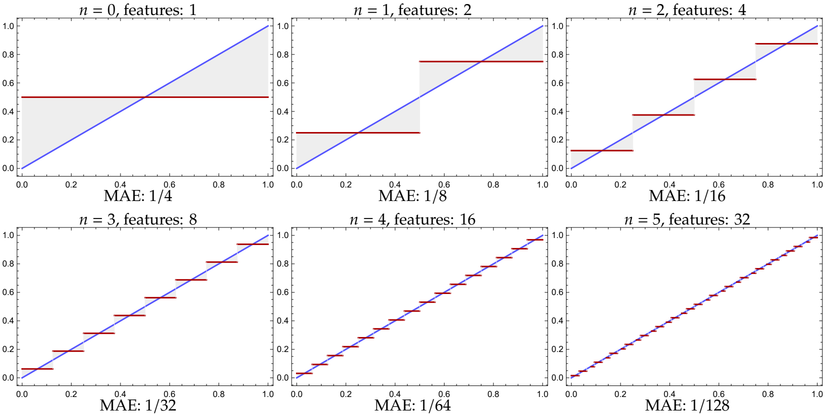
Approximations by Haar wavelets

- Example 1: $f(x) = x$.
- Coefficients:

$$
\begin{aligned}
c_0 &= \frac{1}{\|1\|^2}\langle f, 1\rangle = \frac{1}{1}\int_0^1 x\,dx = \frac{x^2}{2}\Big|_0^1 = \frac{1}{2}\\
c_{1,0} &= \frac{1}{\|\psi_{1,0}\|^2}\langle f, \psi_{1,0}\rangle = \frac{1}{1}\left(\int_{0/2}^{1/2} x\,dx + \int_{1/2}^{2/2} (-x)\,dx\right) = \frac{x^2}{2}\Big|_{0/2}^{1/2} - \frac{x^2}{2}\Big|_{1/2}^{2/2} = -\frac{1}{4}\\
c_{2,0} &= \frac{1}{\|\psi_{2,0}\|^2}\langle f, \psi_{2,0}\rangle = \frac{1}{1/2}\left(\int_{0/4}^{1/4} x\,dx + \int_{1/4}^{2/4} (-x)\,dx\right) = \frac{1}{1/2}\left(\frac{x^2}{2}\Big|_{0/4}^{1/4} - \frac{x^2}{2}\Big|_{1/4}^{2/4}\right) = -\frac{1}{8}\\
c_{2,1} &= \frac{1}{\|\psi_{2,1}\|^2}\langle f, \psi_{2,1}\rangle = \frac{1}{1/2}\left(\int_{2/4}^{3/4} x\,dx + \int_{3/4}^{4/4} (-x)\,dx\right) = \frac{1}{1/2}\left(\frac{x^2}{2}\Big|_{2/4}^{3/4} - \frac{x^2}{2}\Big|_{3/4}^{4/4}\right) = -\frac{1}{8}\\
c_{3,0} &= \frac{1}{\|\psi_{3,0}\|^2}\langle f, \psi_{3,0}\rangle = \frac{1}{1/4}\left(\int_{0/8}^{1/8} x\,dx + \int_{1/8}^{2/8} (-x)\,dx\right) = \frac{1}{1/4}\left(\frac{x^2}{2}\Big|_{0/8}^{1/8} - \frac{x^2}{2}\Big|_{1/8}^{2/8}\right) = -\frac{1}{16}\\
c_{3,1} &= \frac{1}{\|\psi_{3,1}\|^2}\langle f, \psi_{3,1}\rangle = \frac{1}{1/4}\left(\int_{2/8}^{3/8} x\,dx + \int_{3/8}^{4/8} (-x)\,dx\right) = \frac{1}{1/4}\left(\frac{x^2}{2}\Big|_{2/8}^{3/8} - \frac{x^2}{2}\Big|_{3/8}^{4/8}\right) = -\frac{1}{16}\\
c_{3,2} &= \frac{1}{\|\psi_{3,2}\|^2}\langle f, \psi_{3,2}\rangle = \frac{1}{1/4}\left(\int_{4/8}^{5/8} x\,dx + \int_{5/8}^{6/8} (-x)\,dx\right) = \frac{1}{1/4}\left(\frac{x^2}{2}\Big|_{4/8}^{5/8} - \frac{x^2}{2}\Big|_{5/8}^{6/8}\right) = -\frac{1}{16}\\
c_{3,3} &= \frac{1}{\|\psi_{3,3}\|^2}\langle f, \psi_{3,3}\rangle = \frac{1}{1/4}\left(\int_{6/8}^{7/8} x\,dx + \int_{7/8}^{8/8} (-x)\,dx\right) = \frac{1}{1/4}\left(\frac{x^2}{2}\Big|_{6/8}^{7/8} - \frac{x^2}{2}\Big|_{7/8}^{8/8}\right) = -\frac{1}{16}\\
&\ldots
\end{aligned}
$$

Approximations by Haar wavelets

Example 1: $f(x) = x$



$n = 0$, features: 1 — MAE: 1/4

$n = 1$, features: 2 — MAE: 1/8

$n = 2$, features: 4 — MAE: 1/16

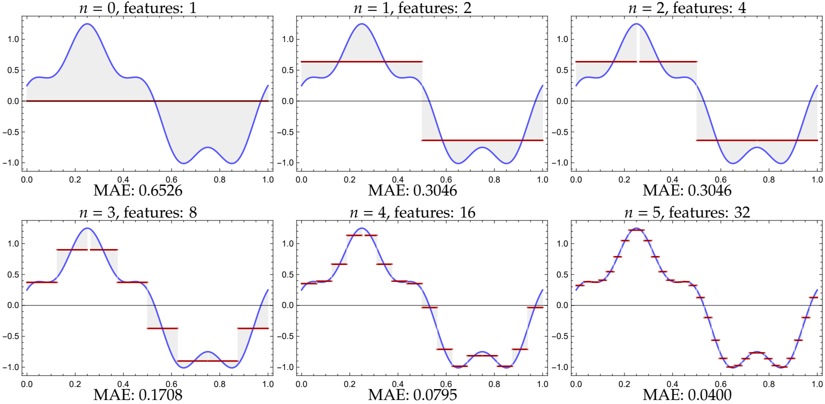
$n = 3$, features: 8 — MAE: 1/32

$n = 4$, features: 16 — MAE: 1/64

$n = 5$, features: 32 — MAE: 1/128

Approximations by Haar wavelets

Example 2: $f(x) = \sin(2\pi x) + 1/4\cos(4 \cdot 2\pi x)$



$n = 0$, features: 1

MAE: 0.6526

$n = 1$, features: 2

MAE: 0.3046

$n = 2$, features: 4

MAE: 0.3046

$n = 3$, features: 8

MAE: 0.1708

$n = 4$, features: 16

MAE: 0.0795

$n = 5$, features: 32

MAE: 0.0400

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Role of expansions in detection / recognition

- Apply **coefficients** of expansions **as input information** (features, attributes) **for detection / recognition** tasks.
- Objects from the same class (faces, people, road signs, etc.) should exhibit certain similarities in their expansion coefficients.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Haar wavelets (2D) — example

- **Orthogonal base generated product-wise**:

$$\psi_{j,k;l,m}(x,y) = \psi_{j,k}(x)\psi_{l,m}(y). \tag{14}$$

- **Wavelet polynomial** (of order n) of two variables:

$$\widehat{f}(x,y) = c_0 \cdot 1 + \sum_{d=1}^{n} \sum_{\substack{0\leqslant j,l\leqslant d\\ j+l=d}} \sum_{k=0}^{2^{j-1}-1} \sum_{m=0}^{2^{l-1}-1} c_{j,k;l,m}\psi_{j,k;l,m}(x,y) \qquad \approx f(x,y), \tag{15}$$

where $(x,y) \in [0,1]\times[0,1]$.

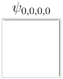
- **Coefficients** (of expansion / approximation):

$$c_{j,k;l,m} = 1/||\psi_{j,k;l,m}||^2\langle f, \psi_{j,k;l,m}\rangle, \qquad c_0 = \langle f, 1\rangle, \tag{16}$$

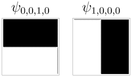
where: $\langle g,h\rangle = \int_0^1 \int_0^1 g(x,y)h(x,y)\,dx\,dy,\ ||g||^2 = \langle g,g\rangle$.

Haar wavelets (2D) — example

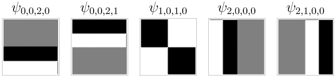
- Term of order 0:



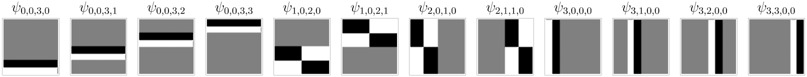
- Terms of order 1:



- Terms of order 2:



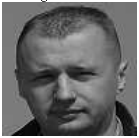
- Terms of order 3:



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Approximations by Haar wavelets (2D)

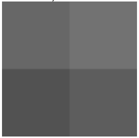
original: 96×96



$n = 0$, features: 1

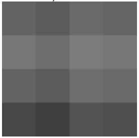
feats/pxs: $1.09 \cdot 10^{-4}$
MAE: 0.1288

$n = 1$, features: 3



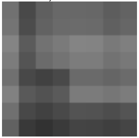
feats/pxs: $3.26 \cdot 10^{-4}$
MAE: 0.1197

$n = 2$, features: 8



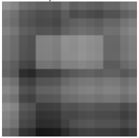
feats/pxs: $8.68 \cdot 10^{-4}$
MAE: 0.1116

$n = 3$, features: 20



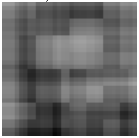
feats/pxs: $2.17 \cdot 10^{-3}$
MAE: 0.0976

$n = 4$, features: 48



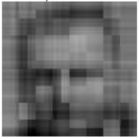
feats/pxs: $5.21 \cdot 10^{-3}$
MAE: 0.0851

$n = 5$, features: 112



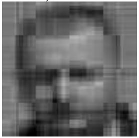
feats/pxs: 0.0122
MAE: 0.0768

$n = 6$, features: 256



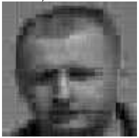
feats/pxs: 0.0278
MAE: 0.0606

$n = 7$, features: 576



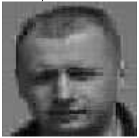
feats/pxs: 0.0625
MAE: 0.0510

$n = 8$, features: 1 280



feats/pxs: 0.1389
MAE: 0.0405

$n = 9$, features: 2 816



feats/pxs: 0.3056
MAE: 0.0324

Fourier moments (2D)

- Consider the following **Fourier approximation**:

$$\widehat{f}(x,y) = \sum_{-n\leqslant j\leqslant n} \sum_{-n\leqslant k\leqslant n} c_{j,k} e^{2\pi i(jx+ky)} \quad \approx f(x,y) \tag{17}$$

  where: $(x,y) \in [0,1]\times[0,1]$ and i is the imaginary unit, $i^2 = -1$.

- **Hermitian inner product**:

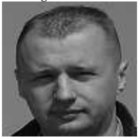
$$\langle g,h\rangle = \int_0^1 \int_0^1 g(x,y)\overline{h(x,y)}\,dx\,dy. \quad \text{(upper bar denotes complex conjugate)} \tag{18}$$

- **Norm**: $\sqrt{\langle g,g\rangle}$ (= 1 for Fourier base).
- **Coefficients — moments**:

$$c_{j,k} = \int_0^1 \int_0^1 f(x,y)e^{-2\pi i(jx+ky)}\,dx\,dy. \tag{19}$$

Approximations by Fourier moments (2D)

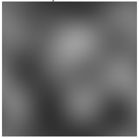
original: 96×96



$n = 0$, features: 1

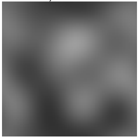
feats/pxs: $1.09 \cdot 10^{-4}$
MAE: 0.1288

$n = 1$, features: 9



feats/pxs: $9.77 \cdot 10^{-4}$
MAE: 0.1100

$n = 2$, features: 25



feats/pxs: $2.71 \cdot 10^{-3}$
MAE: 0.0923

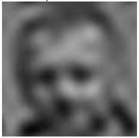
$n = 3$, features: 49

feats/pxs: $5.32 \cdot 10^{-3}$
MAE: 0.0795

$n = 4$, features: 81

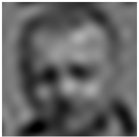
feats/pxs: $8.79 \cdot 10^{-3}$
MAE: 0.0711

$n = 5$, features: 121



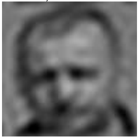
feats/pxs: 0.0131
MAE: 0.0600

$n = 6$, features: 169



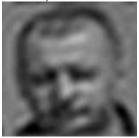
feats/pxs: 0.0183
MAE: 0.0554

$n = 7$, features: 225



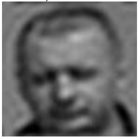
feats/pxs: 0.0244
MAE: 0.0506

$n = 8$, features: 289



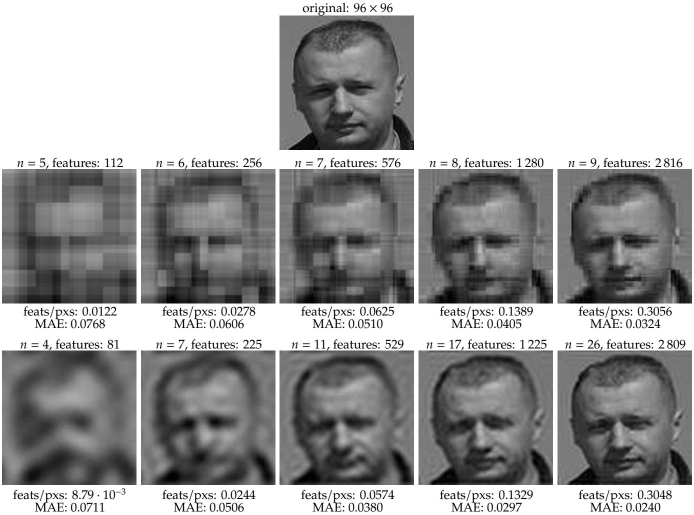
feats/pxs: 0.0314
MAE: 0.0451

$n = 9$, features: 361



feats/pxs: 0.0392
MAE: 0.0417

Approximations: Haar (2D) vs. Fourier (2D)



original: 96 × 96

$n = 5$, features: 112 — feats/pxs: 0.0122, MAE: 0.0768

$n = 6$, features: 256 — feats/pxs: 0.0278, MAE: 0.0606

$n = 7$, features: 576 — feats/pxs: 0.0625, MAE: 0.0510

$n = 8$, features: 1 280 — feats/pxs: 0.1389, MAE: 0.0405

$n = 9$, features: 2 816 — feats/pxs: 0.3056, MAE: 0.0324

$n = 4$, features: 81 — feats/pxs: $8.79 \cdot 10^{-3}$, MAE: 0.0711

$n = 7$, features: 225 — feats/pxs: 0.0244, MAE: 0.0506

$n = 11$, features: 529 — feats/pxs: 0.0574, MAE: 0.0380

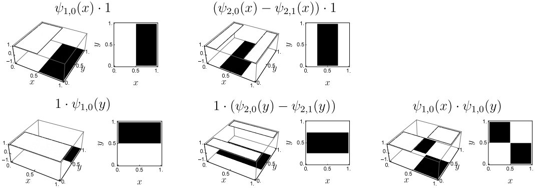
$n = 17$, features: 1 225 — feats/pxs: 0.1329, MAE: 0.0297

$n = 26$, features: 2 809 — feats/pxs: 0.3048, MAE: 0.0240

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Haar-like features (Viola & Jones, 2001)

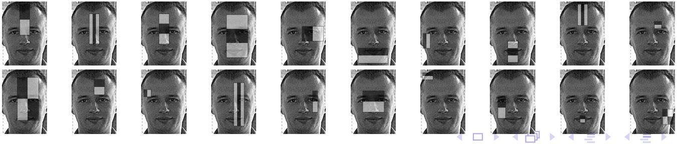
- **Two-dimensional wavelet templates**:



$\psi_{1,0}(x)\cdot 1$ $\quad(\psi_{2,0}(x)-\psi_{2,1}(x))\cdot 1$

$1\cdot\psi_{1,0}(y)$ $\quad 1\cdot(\psi_{2,0}(y)-\psi_{2,1}(y))$ $\quad\psi_{1,0}(x)\cdot\psi_{1,0}(y)$

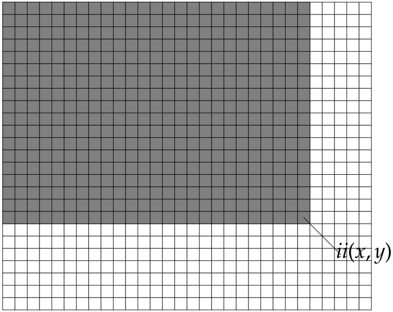
- Templates mapped to **features** by scaling and anchoring within detection window (orthogonality can be neglected).
- Features: **differences in averages of pixel intesities** under white and black regions (rough contours).
- Intention: **“brute force attack on features”** — to generate a great multitude e.g. $\sim 10^5$.
- Some of features might happen to represent good characteristics of targets.



Integral image (repeated)

- Image function: $i(x, y)$ — pixel intensity at (x, y)
- **Integral image** $ii(x, y)$ defined as:

$$ii(x, y) = \sum_{1\leqslant j\leqslant x} \sum_{1\leqslant k\leqslant y} i(j, k). \tag{20}$$

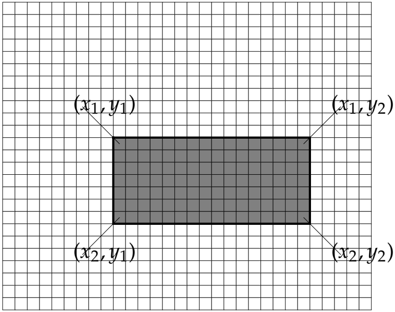


Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Integral image (repeated)

- *How (having prepared ii) to calculate the sum of intensities in a rectangle spanning from* (x_1, y_1) *to* (x_2, y_2)*?*

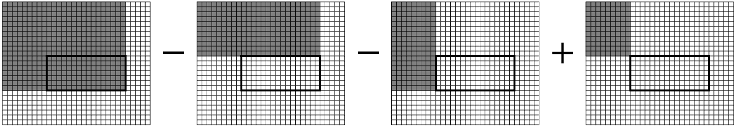
$$\sum_{x_1 \leqslant x \leqslant x_2} \sum_{y_1 \leqslant y \leqslant y_2} i(x,y) = ? \tag{21}$$



Growth of integral image

- Constant-time calculation of a sum:

$$\sum_{x_1 \leqslant x \leqslant x_2} \sum_{y_1 \leqslant y \leqslant y_2} i(x,y) = ii(x_2,y_2) - ii(x_1-1,y_2) - ii(x_2,y_1-1) + ii(x_1-1,y_1-1). \qquad (22)$$



- **Sufficient are 3 operations** on 4 points read from integral image array regardless of rectangle size — $O(1)$.
- Analogy to calculus (growth of antiderivative / primitive function):

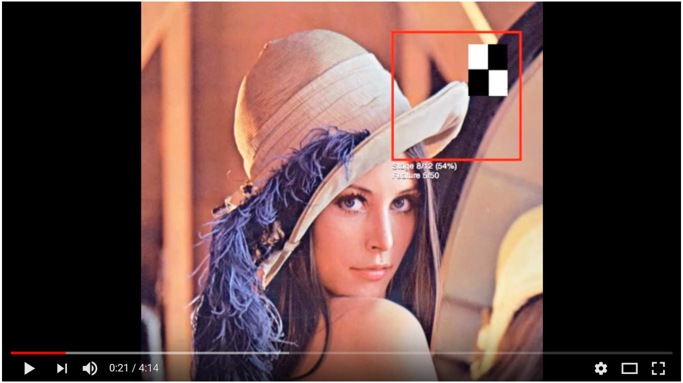
$$\int_{x_1}^{x_2} \int_{y_1}^{y_2} f(x,y)\,dx\,dy = F(x_2,y_2) - F(x_1,y_2) - F(x_2,y_1) + F(x_2,y_2), \qquad (23)$$

where F is antiderivative for f, that is: $F(x,y) = \int_{-\infty}^{x} \int_{-\infty}^{y} f(u,v)\,du\,dv$.

- **‘edge features’** (Haar-like): **8** or **9** operations, **‘diagonal features’**: **13** operations.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Vizualization



[by Adam Harvey, YouTube: https://www.youtube.com/watch?v=hPCTwxF0qf4]

Number of features — parameterization

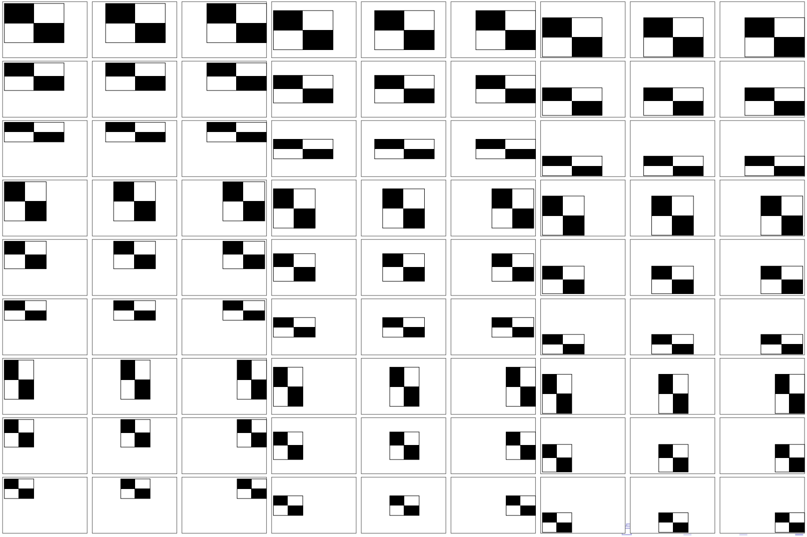
- Commonly, some parameterization is introduced using: **scaling** and **positioning** on templates within window.
- Let q denote the number of possible scalings along one dimension. Hence, there exist q^2 scaled versions for each template.
- Let p generate a regular grid $(2p-1)\times(2p-1)$ of anchoring points for features.
- **Total number of features**:

$$n(q,p) = 5q^2(2p-1)^2. \tag{24}$$

- Examples:

	$p=1$	$p=2$	$p=3$	$p=4$	$p=5$
$q=1$	5	45	125	245	405
$q=2$	20	180	500	980	1 620
$q=3$	45	405	1 125	2 205	3 645
$q=4$	80	720	2 000	3 920	6 480
$q=5$	125	1 125	3 125	6 125	10 125

Example of parameterization for: $q = 3, p = 2$



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

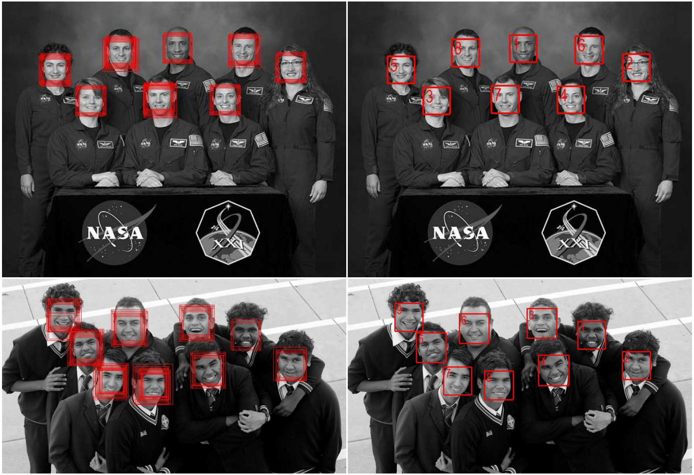
Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

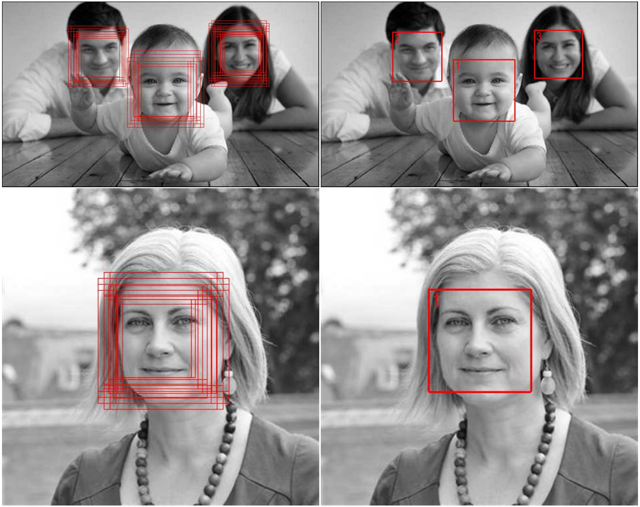
Experimental setup

- **Train data**: 7 258 positive examples, 100 000 negative examples.
- **Learning algorithm**: **RealBoost + bins**, ensemble sizes: $T = 256$ or $T = 512$.
- **Test data**: $\approx$ 70 500 000 windows within 500 images containing 1 000 faces.
- To conveniently generate **ROCs** a test *subset* generated with $2 \cdot 10^6$ negatives → precision along FAR axis: $5 \cdot 10^{-7}$.
- **Feature spaces**:
  (1) $q = 3, p = 3$ (1 125 feats.),
  (2) $q = 4, p = 3$ (2 000 feats.),
  (3) $q = 3, p = 4$ (2 205 feats.),
  (4) $q = 4, p = 4$ (3 920 feats.),
  (5) $q = 5, p = 5$ (10 125 feats.).
- Train data sizes: from 0.5 GB to 4.3 GB
- **Detection procedure 1 ("heavy"):** $\approx$ 151 000 windows
  (8 scales, sliding window 48 × 48 up to 172 × 172, jumps ratio 0.05).
- **Detection procedure 2 ("light"):** $\approx$ 11 000 windows
  (4 scales, sliding window 120 × 120 up to 207 × 207, jumps ratio 0.05)
- Software written in **C#** with key procedures in **C++** as dll libraries.

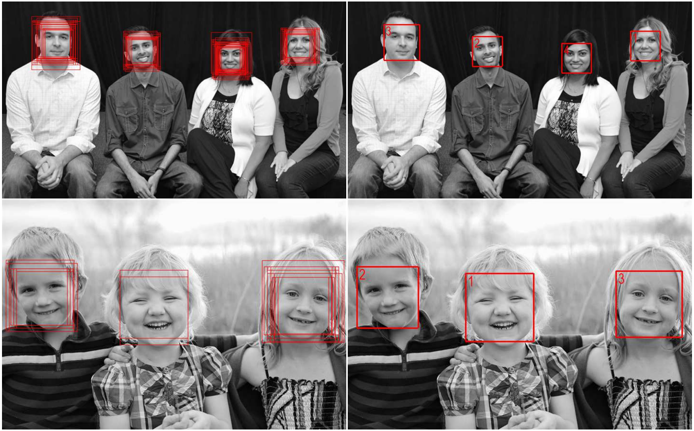
Examples of outcomes



Examples of outcomes

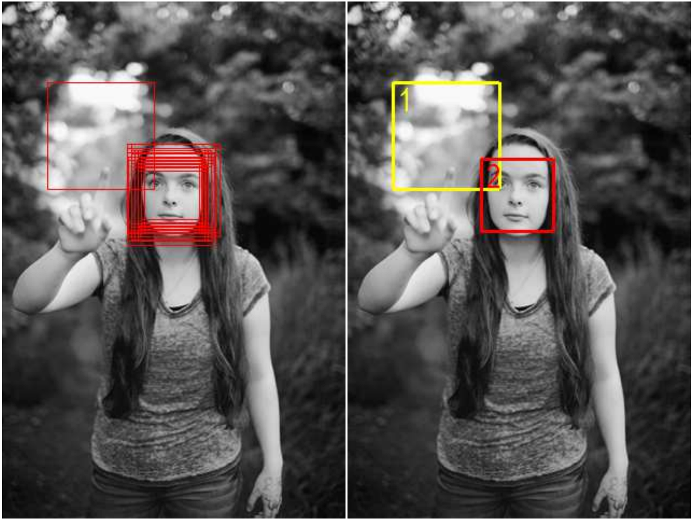


Examples of outcomes



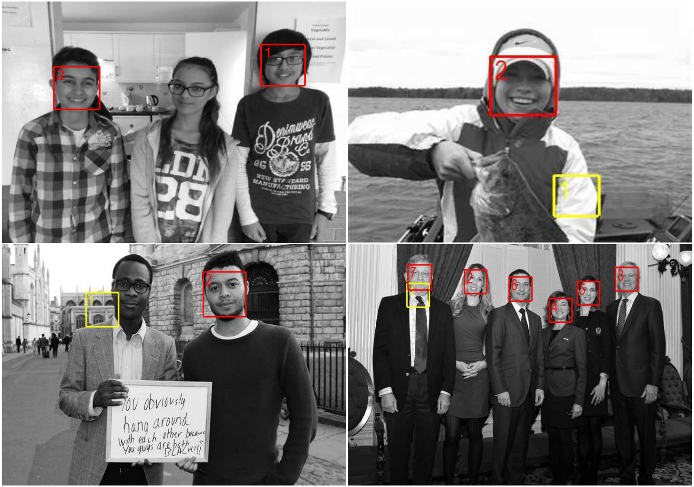
Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Examples of outcomes (with errors)

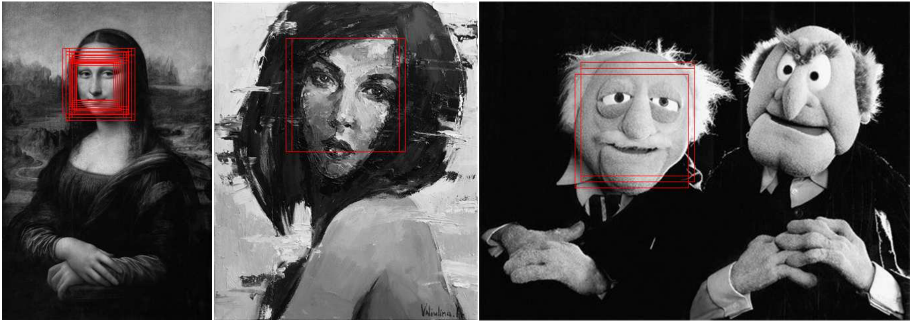


Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

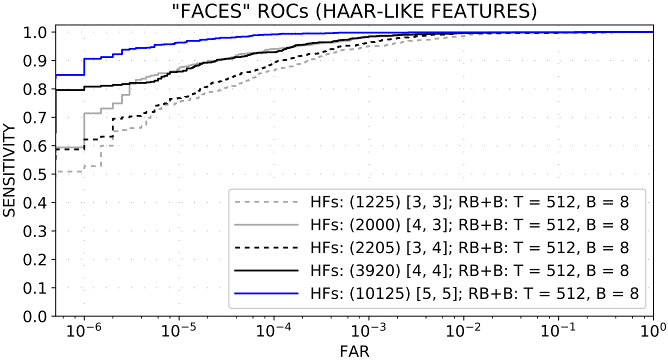
Examples of outcomes (with errors)


2
1
2
1
1
2
YOU obviously hang around with each other because you guys are both BLACK!!!
7
1
2
6
4
5
3

False alarms or faces?



ROC curves


"FACES" ROCs (HAAR-LIKE FEATURES)
SENSITIVITY
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
FAR
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
HFs: (1225) [3, 3]; RB+B: T = 512, B = 8
HFs: (2000) [4, 3]; RB+B: T = 512, B = 8
HFs: (2205) [3, 4]; RB+B: T = 512, B = 8
HFs: (3920) [4, 4]; RB+B: T = 512, B = 8
HFs: (10125) [5, 5]; RB+B: T = 512, B = 8

Accuracy measures

name / description	AUC$_\alpha$			sensitivity	FAR per image	FAR per window	accuracy per window
	α=10^{-5}	α=10^{-4}	α=10^{-3}				
HF q=3, p=3 (1125); T=512	0.6761	0.8123	0.9156	0.699	0.098	$6.975\cdot10^{-7}$	0.999995018084708
HF q=4, p=3 (2000); T=512	0.8021	0.9082	0.9624	0.849	0.086	$6.121\cdot10^{-7}$	0.999997238589628
HF q=3, p=4 (2205); T=512	0.7075	0.8376	0.9320	0.741	0.084	$5.978\cdot10^{-7}$	0.999995715550288
HF q=4, p=4 (3920); T=512	0.8188	0.9141	0.9729	0.897	0.102	$7.234\cdot10^{-7}$	0.999997815602899
HF q=5, p=5 (10125); T=512	0.9353	0.9793	0.9951	0.970	0.066	$4.681\cdot10^{-7}$	0.999999106383004

Time performance

quantity (or operations)	**“heavy” procedure** Haar-like features ($T = 512$) (495 distinct feats.)	**“light” procedure** Haar-like features ($T = 512$) (495 distinct feats.)
no. of analyzed windows	151 385	11 838
preparation time for integral image	6 ms	6 ms
total time of detection procedure	513 ms	83 ms
time per 1 window	3.38 μs (amortized: 3.34 μs)	7.01 μs (amortized: 6.50 μs)
time per 1 window and 1 feature	6.83 ns (amortized: 6.75 ns)	14.16 ns (amortized: 13.14 ns)

[640 × 480 image; parallel computations on: Intel Xeon E3-1505M v5 4×2-core 2.80 (3.70) GHz CPU;]

[cascade of classifiers *not* used]

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Table of contents

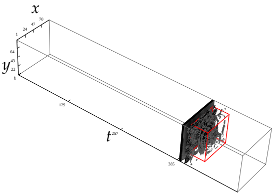
1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

Landmine detector (3D HFs)

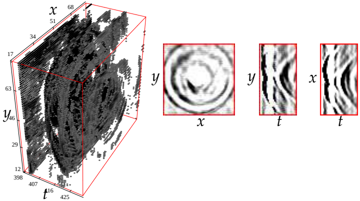
- 3D images — **C-scans** — from **GPR (Ground Penetrating Radar)**.
- Coordinates: *across track* × *along track* × *time*. Image function: $i(x, y, t)$.
- Time axis can be inuitively associated with depth. Radar working in frequency domain — time samples obtained from complex signals via IFFT.
- Objects non-transparent to GPR generated **hiperboloids** in images.
- **R&D project** by: (Olech, Kapruziak, Godziuk, Klęsk, 2011–2014). In particular, research on various **3D features computed via integral images**: Haar-like features, statistical moments, Fourier moments, HOG descriptor.

scene before burial

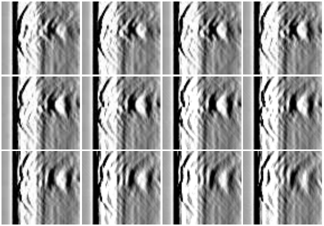
C-scan (thresholded)
detection window at
$(x, y, t) = (17, 12, 398)$



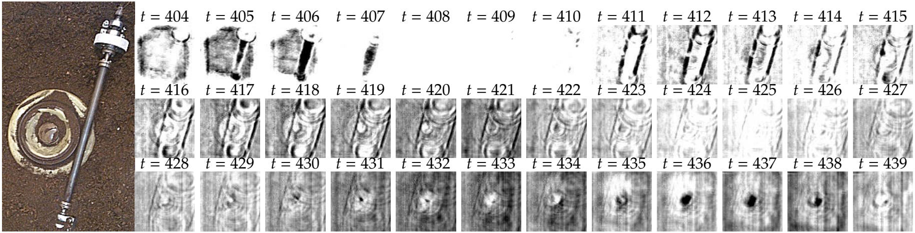
close up and slices



Landmine detector (3D HFs)



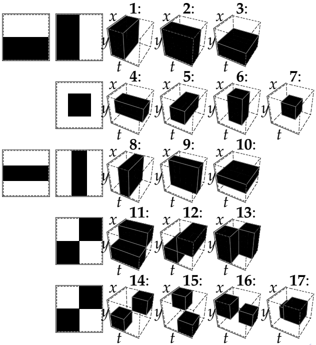
Landmine detector (3D HFs)



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

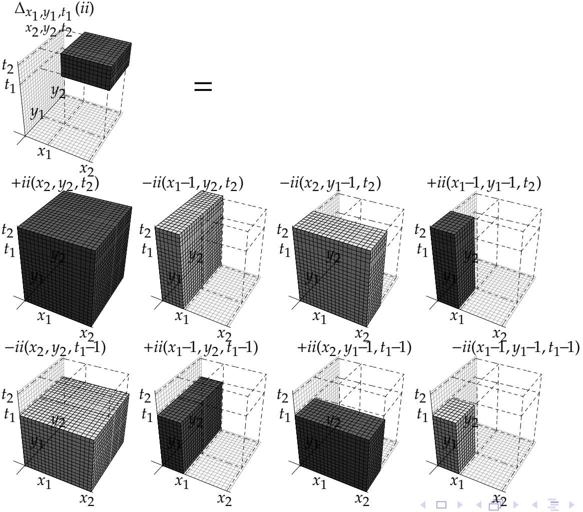
Landmine detector (3D HFs)

- Proposition of **17 templates** for 3D Haar-like features.
- **No. of features** at learning stage: **17 000**.
- **Train data** based on **210 C-scans**: ≈ 7 GB (≈ 100 000 examples of 3D windows).
- **Learning algorithm**: **boosted decision trees** (shallow trees — 4 or 8 terminals).
- Final calssifier (ensemble) consisting of **600 trees** and thereby at most 1 800 or 4 200 features (for 4 and 8 terminals, respectively).



Landmine detector (3D HFs)

- Growth of integral image $ii(x, y, t)$ based on 8 points:



$\Delta_{x_1,y_1,t_1}(ii)$ x_2,y_2,t_2 $=$

$+ii(x_2, y_2, t_2)$ $-ii(x_1{-}1, y_2, t_2)$ $-ii(x_2, y_1{-}1, t_2)$ $+ii(x_1{-}1, y_1{-}1, t_2)$

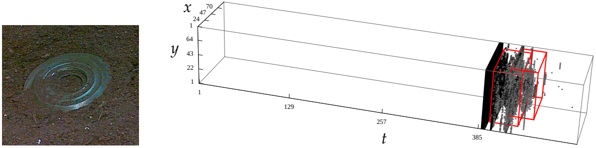
$-ii(x_2, y_2, t_1{-}1)$ $+ii(x_1{-}1, y_2, t_1{-}1)$ $+ii(x_2, y_1{-}1, t_1{-}1)$ $-ii(x_1{-}1, y_1{-}1, t_1{-}1)$

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

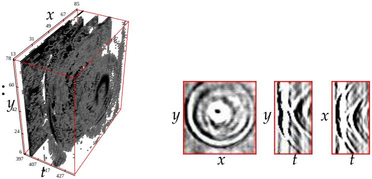
Landmine detector (3D HFs)

- Example of metal AT mine detection with a side false alarm:

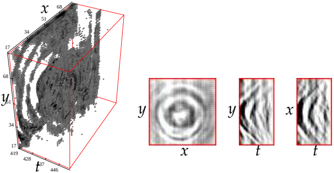
1:



1.1:



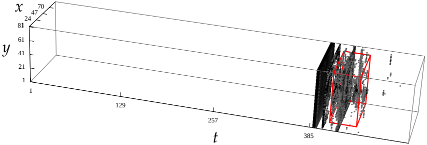
1.2:



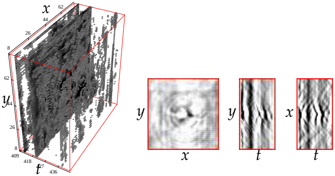
Landmine detector (3D HFs)

- Example of plastic AT mine detection:

2:

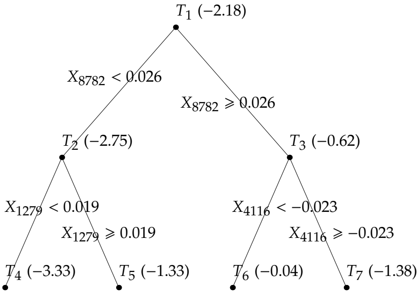


2.1:



Landmine detector (3D HFs)

- Examples of relevant features in the first decision tree (within ensemble):



T_1 (-2.18)

$X_{8782} < 0.026$

$X_{8782} \geqslant 0.026$

T_2 (-2.75)

T_3 (-0.62)

$X_{1279} < 0.019$

$X_{1279} \geqslant 0.019$

$X_{4116} < -0.023$

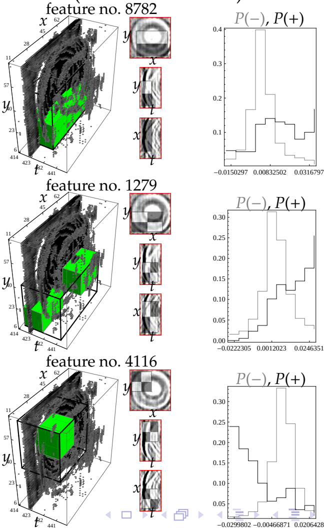
$X_{4116} \geqslant -0.023$

T_4 (-3.33)

T_5 (-1.33)

T_6 (-0.04)

T_7 (-1.38)



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Landmine detection — selected papers

- **P. Klęsk, M. Kapruziak, and B. Olech**, "Fast Extraction of 3D Fourier Moments via Multiple Integral Images: An Application to Antitank Mine Detection in GPR C-Scans" in *International Conference on Computer Vision and Graphics (ICCVG)*, 2016, pp. 206–220.
- **P. Klęsk, M. Kapruziak, and B. Olech**, "Statistical moments calculated via integral images in application to landmine detection from Ground Penetrating Radar 3D scans", *Pattern Analysis and Applications*, 2017, pp. 1–14.
- **P. Klęsk, A. Godziuk, M. Kapruziak, and B. Olech**, "Fast analysis of C-scans from ground penetrating radar via 3-D Haar-like features with application to landmine detection", ***IEEE Transactions on Geoscience and Remote Sensing***, vol. 53, no. 7, 2015, pp. 3996–4009.
- **P. Klęsk, M. Kapruziak, and B. Olech**, "A Comparison of Shallow Decision Trees Under Real-Boost Procedure with Application to Landmine Detection Using Ground Penetrating Radar" in *International Conference on Artificial Intelligence and Soft Computing (ICAISC)*, 2015, pp. 436–447.

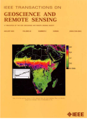


Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

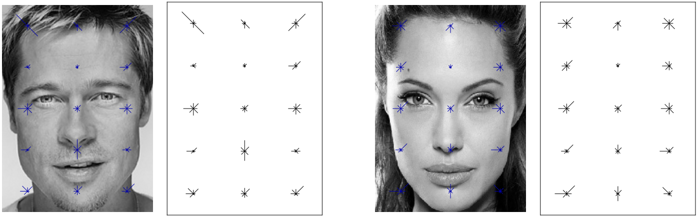
Histograms of Oriented Gradients

- Idea described first in (Dalal & Triggs, 2005).
- The technique observes **orientations of local gradients** present within **cells** of image windows.
- Gradient orientations (within $[-\pi/2, \pi/2]$ or $[0, 2\pi]$) are **dicretized**.
- Detection window partitioned into a **regular grid of cells**.
- **Each pixel "votes"** within its cell for some orientation of gradient with vote strength proportional to gradient magnitude anchored at that pixel.
- Cells are grouped into larger **blocks** for **normalization** and mitigation of local image constrasts.
- **Feature vector**: concatenation of **gradient distributions over all cells**.

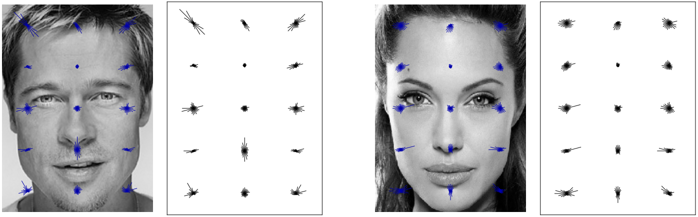
Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

HOG — examples for faces

- Discretization of $[0, 2\pi]$ into $n_\theta = 8$ intervals. Grid of cells: 5×3. Features: 120.

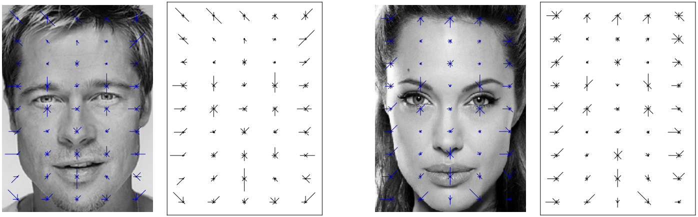


- Discretization of $[0, 2\pi]$ into $n_\theta = 24$ intervals. Grid of cells: 5×3. Features: 360.

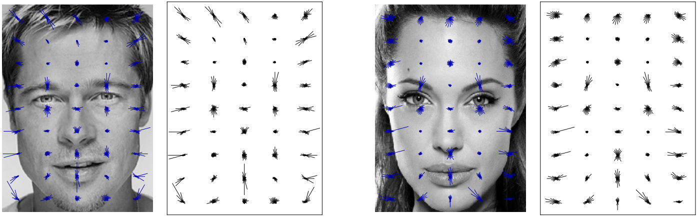


HOG — examples for faces

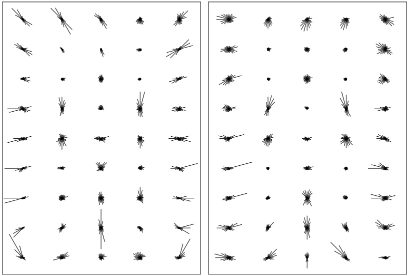
- Discretization of $[0, 2\pi]$ into $n_\theta = 8$ intervals. Grid of cells: 9×5. Features: 360.



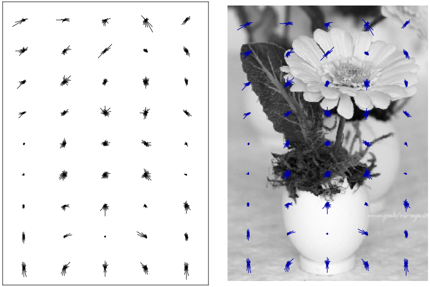
- Discretization of $[0, 2\pi]$ into $n_\theta = 24$ intervals. Grid of cells: 9×5. Features: 1 080.



HOG — face vs. non-face

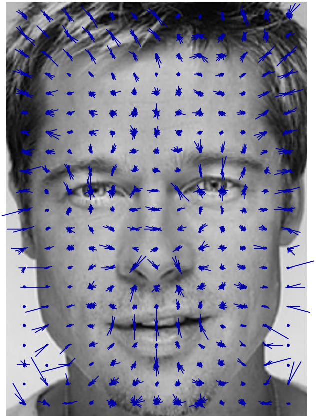


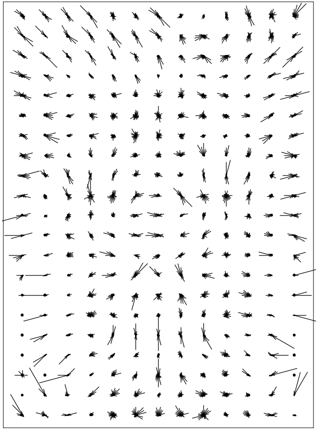
vs.



HOG — examples for faces

- Discretization of $[0, 2\pi]$ into $n_\theta = 24$ intervals. Grid: 21×13. Features: 6 552.
- Visualization on a dense grid gradually resembles a face.

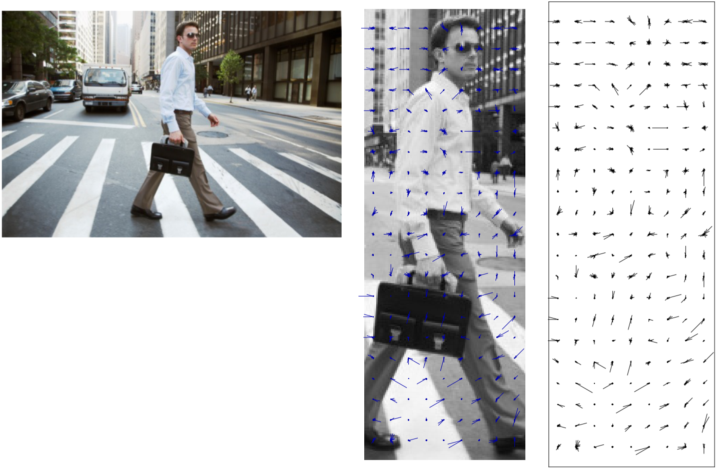




Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

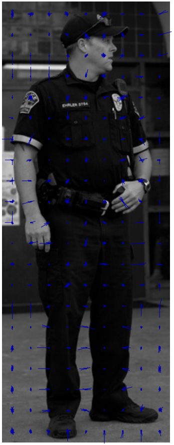
HOG — examples for pedestrians

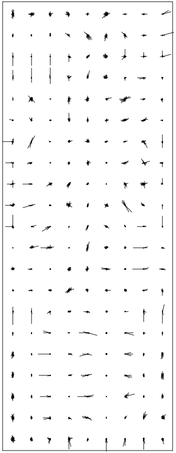
- Discretization of $[0, 2\pi]$ into $n_\theta = 24$ intervals. Grid: 21×9. Features: 4 536.



HOG — examples for pedestrians

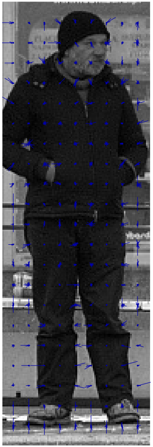
- Discretization of $[0, 2\pi]$ into $n_\theta = 24$ intervals. Grid: 21×9. Features: $4\,536$.

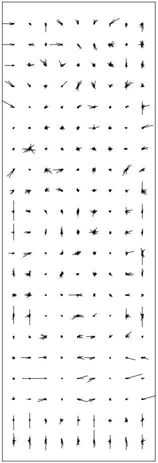




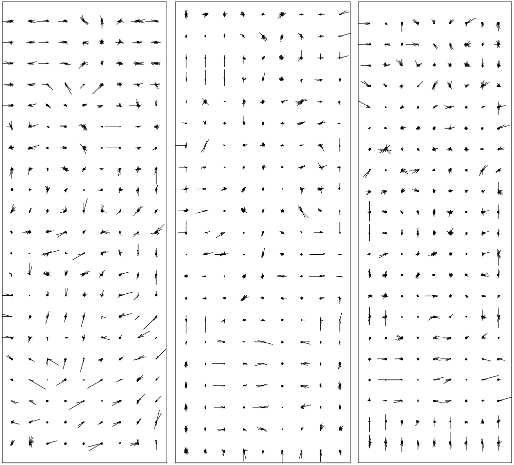
HOG — examples for pedestrians

- Discretization of $[0, 2\pi]$ into $n_\theta = 24$ intervals. Grid: 21×9. Features: $4\,536$.

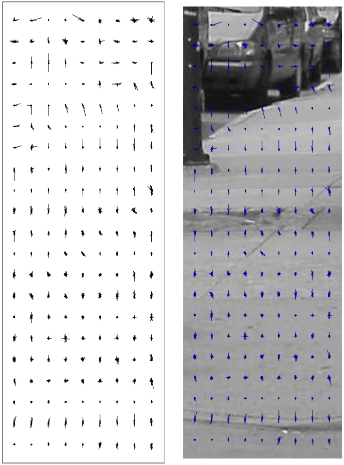




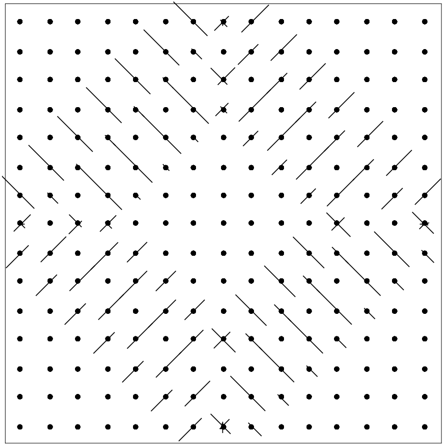
HOG — pedestrian vs. non-pedestrian



vs.

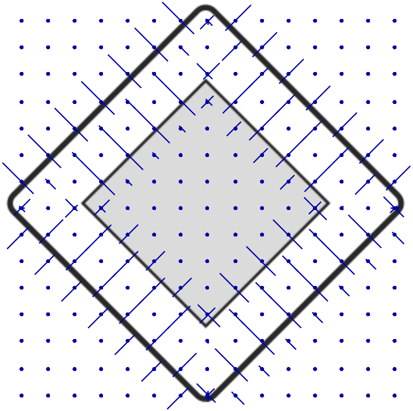


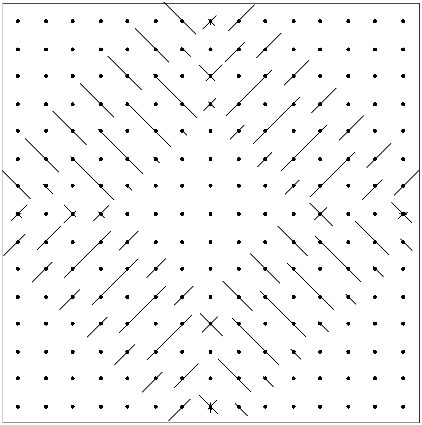
HOG — what is this?



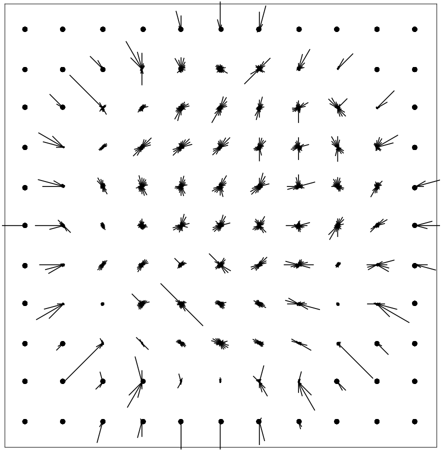
Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

HOG — what is this?

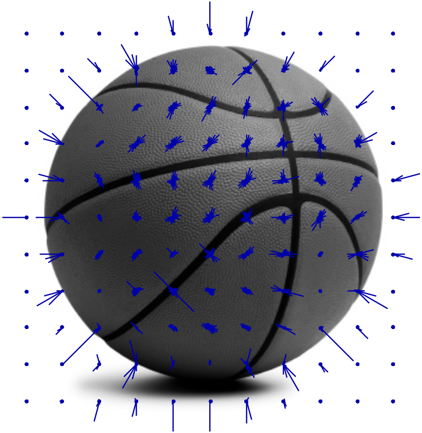


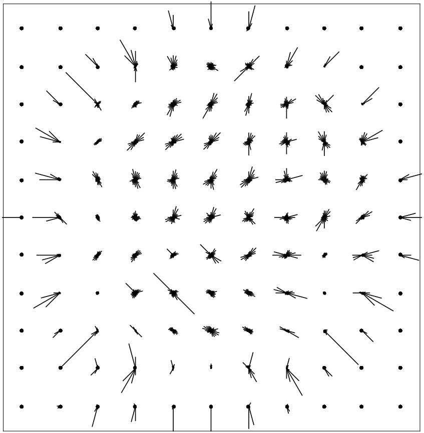


HOG — what is this?



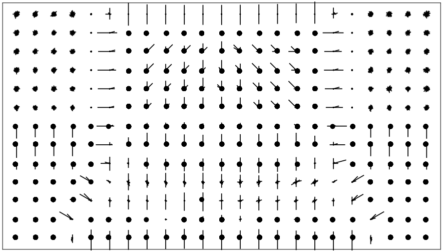
HOG — what is this?



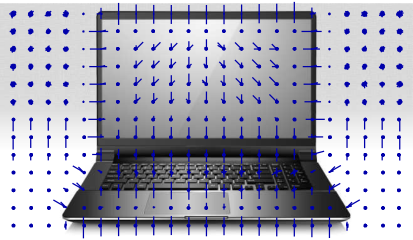


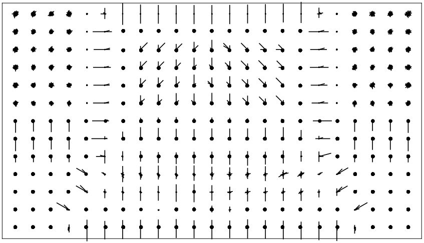
HOG — what is this?

?



HOG — what is this?





HOG — how it works (1)

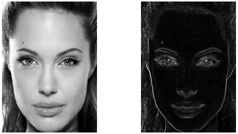
- Convert image to **grayscale**.
- **Convolve image with simple gradient filters**: $h_x = (-1, 0, 1)$, $h_y = (-1, 0, 1)^T$:

$$g_x = i * h_x \tag{25}$$

$$g_y = i * h_y. \tag{26}$$

- Calculate **gradient magnitude** at each pixel (j, k) as:

$$G(j,k) = \sqrt{{g_x}^2(j,k) + {g_y}^2(j,k)}. \tag{27}$$



HOG — how it works (2)

- For each pixel find **dominating angle** $\theta(j,k)$.
- There exist **two possibilities of angle range** to consider: $[-\pi/2, \pi/2]$ or $[0, 2\pi)$.
- Depends on whether we want to take into account or neglect where gradients are headed.
- True gradients, within $[0, 2\pi)$, are headed from darker to lighter regions.
- Angular ranges are implied by the choice of **tangent arc** function, i.e. $\tan^{-1}$:

$$\theta(j,k) = \mathrm{atan}(g_y(j,k)/g_x(j,k)), \tag{28}$$

or

$$\theta(j,k) = \mathrm{atan2}(g_y(j,k), g_x(j,k)). \tag{29}$$

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

HOG — how it works (3)

- **Individual values** (for single pixels) of angles $\theta(j,k)$ and magnitudes $G(j,k)$ can be **strongly variable**, even for similar images.
- Therefore, **aggregations** of $\theta(j,k)$ and $G(j,k)$ are introduced over some rectangular surroundings — **cells**.
- This leads to a more stable description — **robustness** to small changes or noises.
- Cell sizes (in pixels) are implied by the grid size.
- Denser grids produce more features but become gradually more susceptible to noises.
- **Angular range** is **discretized** into an imposed number n_θ of equally wide intervals — **bins**.
- **Each pixel "votes"** for the bin to which its dominating angle $\theta(j,k)$ belongs, **proportionally to gradient magnitude**: $G(j,k)$.
- **Normalized sums of votes** for particular cells form stable statistics, and thereby features.

HOG — how it works (4)

- Let border angles be defined as:

$$\phi_l = -\pi/2 + l\pi/n_\theta, \qquad l = 0, 1, \ldots, n_\theta; \tag{30}$$

$$\phi_l = -\pi/n_\theta + l2\pi/n_\theta, \qquad l = 0, 1, \ldots, n_\theta; \tag{31}$$

  respectively for $[-\pi/2, \pi/2]$ and $[0, 2\pi)$.

- Hence, middle angles (representatives) in particular bins are:

$$(\phi_l + \phi_{l-1})/2, \qquad l = 1, \ldots, n_\theta. \tag{32}$$

- In case of $[0, 2\pi)$ range, the middle angle for the first bin coincides with horizontal axis.
- “Circularity” of the angular axis should be taken into account (i.e: $-\pi/n_\theta$ corresponds to $2\pi - \pi/n_\theta$).

HOG — how it works (5)

- A **matrix of votes** V of size $n_x \times n_y \times n_\theta$ is formulated:

$$V(j,k,l) = \begin{cases} G(j,k), & \text{when } \phi_{l-1} \leqslant \theta(j,k) < \phi_l; \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

- Votes are summed and memorized seperately for each pair of cell c and bin ($l = 1, \ldots, n_\theta$):

$$H_1(c,l) = \sum\nolimits_{(j,k)\in c} V(j,k,l). \tag{34}$$

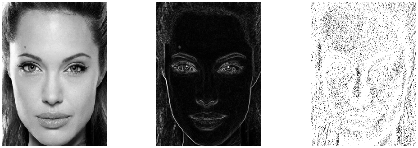
- Final **features** $H(c,l)$ of HOG descriptor are calculated from H_1 values by performing their **normalization over blocks of cells**, i.e. cells being direct neighbours:

$$H(c,l) = H_1(c,l) \Big/ \sum\nolimits_{c_q \in N(c)} \sqrt{\|H_1(c_q)\|_2^2 + \epsilon^2}, \tag{35}$$

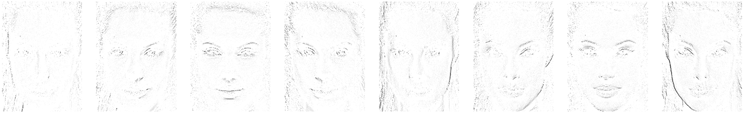
where: $N(c)$ denotes the set of neighbours for a cell c, $H_1(c) = \big(H_1(c,1), \ldots, H_1(c,n_\theta)\big)$, $\epsilon > 0$ is a selectable constant, and $\|\cdot\|$ denotes Euclidean norm.

HOG — visualization of successive steps

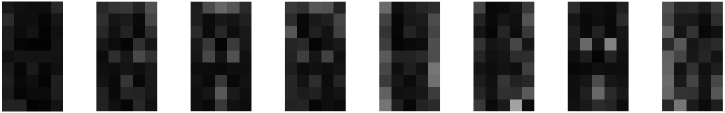
- Original image, image with gradients $G(j,k)$, and image with angles $\theta(j,k)$:



- Images of votes for particular bins (example for $n_\theta = 8$)[1]:



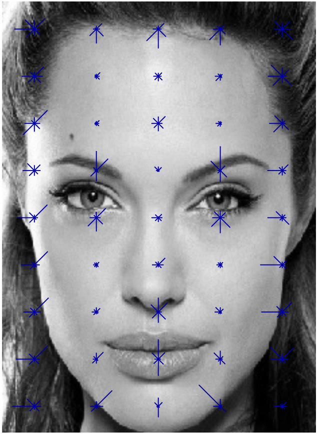
- Feature values $H(c,l)$ over 9×5 grid ($n_\theta = 8$):



[1] For readability, imaging with negation of gray levels and sharpening.

HOG — visualization of successive steps

- Final visualization — gradients of lengths $H(c, l)$ are drawn along representative angles at cell centers:



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

HOG — integral images

- Question: *Which computational step could be speeded up by integral images within a detection procedure?*

HOG — integral images

- **Answer: sums of votes within cells**:

$$H_1(c,l) = \sum_{(j,k)\in c} V(j,k,l).$$

- For n_θ bins, one should introduce a set of n_θ **integral images to cumulate votes**:

$$ii_l(x,y) = \sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} V(j,k,l), \quad l = 1,\ldots,n_\theta. \tag{36}$$

- Value $H_1(c,l)$ for a cell c, spanning from $(x_1(c),y_1(c))$ to $(x_2(c),y_2(c))$, can be then calculated as:

$$H_1(c,l) = ii_l\left(x_2(c),y_2(c)\right) - ii_l\left(x_1(c){-}1,y_2(c)\right) - ii_l\left(x_2(c),y_1(c){-}1\right) + ii_l\left(x_1(c){-}1,y_2(c){-}1\right). \tag{37}$$

- **Owing to integral images, extraction of each HOG feature becomes a constant-time — O(1) — operation.**

Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. **Boosting — variants and properties**
8. Cascades of classifiers
9. Towards new results

Boosting as a meta algorithm

- **Sketch of idea** appeared in paper: *"The strength of weak learnability"* **(Schapire, 1990)**.
- Successive important works, shaping the current form of boosting, were: (Freund, 1995; Freund & Schapire, 1996, 1997; Friedman, Hastie, & Tibshirani, 2000; Schapire & Singer, 1999).
- Boosting **applies sequentially a simple learning algorithm on reweighted data** — each training example has a *weight* which changes during successive boosting rounds.
- In effect, we obtain an **ensemble** of **partial classifiers**, also referred to as **weak classifiers** — *"anything better than a coin toss will do"*.
- **Final response** of ensemble classifier for some input object is calculated as a **majority vote** or a **weighted sum** from responses of weak classifiers.
- Boosting algorithms appear to be **well-suited** for **large data sets**.
- Important properties observed in practice:
  (1) **capability to automatically select relevant features**,
  (2) **robustness to overfitting** — as new weak classifiers are added to ensemble, test error stabilizes (instead of increasing).
- It can be demonstrated mathematically that boosting can be seen as an **sequential additive model** for **logistic regression**.

Notation

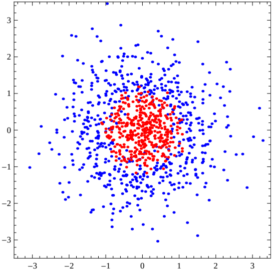
- Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,m}$ denote the set of training examples, where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ are feature vectors, and $y_i \in \{-1, 1\}$ are class labels.
- Rounds (iterations) of boosting procedure shall be numbered as $t = 1, 2, \ldots, T$.
- Let w_i denote the weight of i-th example on current boosting round.
- In case it is needed, we shall write $w_{i,t}$ to indicate the round index explicitly.
- Weigths can be regarded as a probability distribution over data examples, i.e.: $w_i \geqslant 0$ and $\sum_{i=1}^{m} w_i = 1$.
- Let f_t denote a weak classifier produced on round t.
- Let F denote the whole ensemble.
- When observing progress of algorithm, let F_t (with subindex) denote the current state of ensemble on round t, i.e. ordinary or weighted sum of $f_1, f_2, \ldots, f_t$. In this sense, notation F_T is equivalent with F.
- Let $[s]$ denote an indicator function, yielding 1 when s is true and 0 otherwise.

Discrete AdaBoost

```
algorithm DiscreteAdaBoost($\mathcal{D}$)
    start with weights: $w_i := 1/m,\ i = 1, \ldots, m$
    for $t := 1, \ldots, T$ repeat
        fit weak classifier $f_t(\mathbf{x}) \in \{-1, 1\}$ using weights $w_i$ on training data
        calculate train error:
            $\epsilon_t := \sum_{i=1}^{m} w_i [f(\mathbf{x}_i) \neq y_i]$
        calculate classifier's coefficient (importance):
            $\alpha_t := \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
        update weights:
            $Z_t := \sum_{i=1}^{m} w_i e^{-\alpha_t f_t(\mathbf{x}_i) y_i}$
            $w_i := w_i e^{-\alpha_t f_t(\mathbf{x}_i) y_i} / Z_t, \qquad i = 1, \ldots, m$
    return ensemble $F(\mathbf{x}) := \sum_{t=1}^{T} \alpha_t f_t(\mathbf{x})$ with decision calculated as $\operatorname{sgn} F(\mathbf{x})$
```

AdaBoost — data for experiments

- Data drawn (i.i.d.) from joint probability distribution: $P(\mathbf{x}, y) = p(\mathbf{x})P(y|\mathbf{x})$.
- $p(\mathbf{x})$ — density of bivariate normal distribution $N^2(0, 1)$.
- Conditional: $P(y|\mathbf{x}) = 1/\left(1 + e^{y\beta({x_1}^2+{x_2}^2-r^2)}\right)$, where $r = 1$, $\beta = 5$.
- Train data ($m = 1000$ examples):



- **True error** for some classifier $c(\mathbf{x}) \in \{-1, 1\}$:

$$\mathrm{err}_P(c) = \int_{-\infty}^{\infty} \sum_{y\in\{-1,1\}} [c(\mathbf{x}) \neq y]\, P(y|x)p(\mathbf{x})\, \mathbf{dx}. \qquad (38)$$

- True error for optimal classifier $c(\mathbf{x}) = 2[{x_1}^2 + {x_2}^2 - r^2 < 0] - 1$ is ≈ 0.084442.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

AdaBoost + decision stumps (1)

- Each weak classifier is based on a single selected feature and performs a thresholded decision:

$$f_t(\mathbf{x}; j, v, d) = \begin{cases} 1, & \text{for } d(x_j - v) > 0; \\ -1, & \text{otherwise;} \end{cases} \tag{39}$$

  where $j \in \{1, \dots, n\}$ — feature index, $v \in \mathbb{R}$ — threshold, and $d \in \{-1, 1\}$ — decision direction.

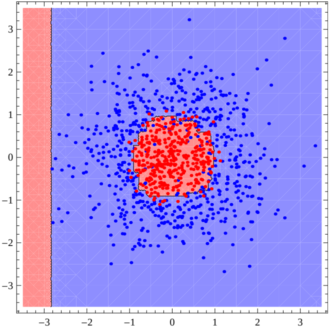
- Selection of a triplet (feature, threshold, direction) is typically carried out by minimization of train error resulting from the split[2]:

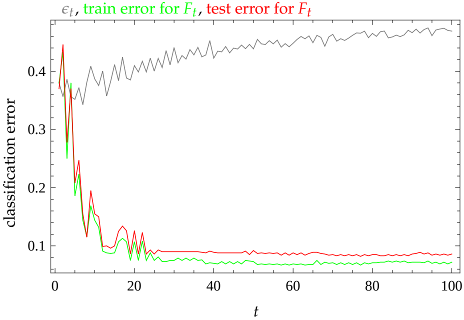
$$(j^*, v^*, d^*) = \arg\min_{(j,v,d)} \sum_{i=1}^{m} w_i [f_t(\mathbf{x}_i; j, v, d) \neq y_i]. \tag{40}$$

[2] Other approaches possible: maximum information gain, minimum Gini index

AdaBoost + decision stumps (2)

- Final decision boundary for the ensemble ($T = 100$) and error plots:

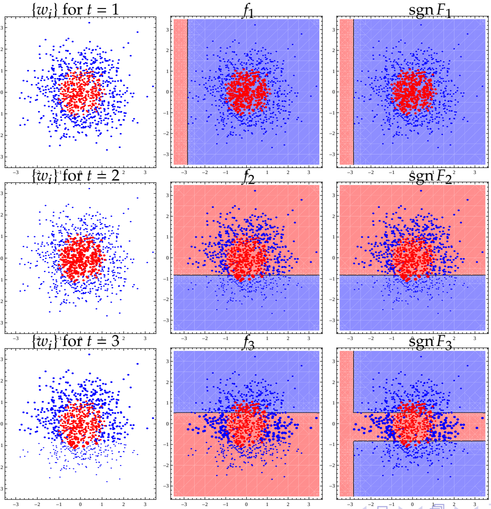




- True error: $\mathrm{err}_P(F) \approx 0.092805$.
- Error of F on test sample (also containing 1000 examples): 0.080.

AdaBoost + decision stumps (3)

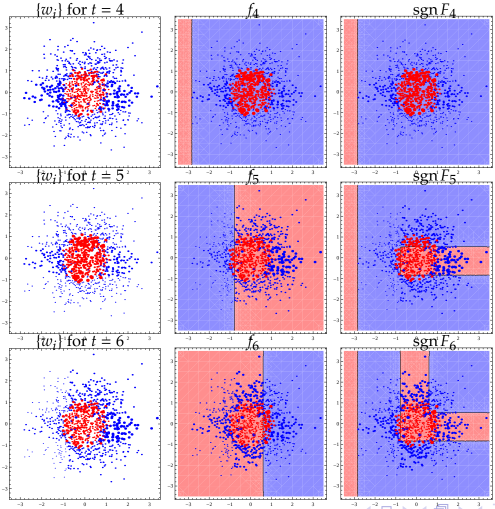
- Learning progress:



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

AdaBoost + decision stumps (4)

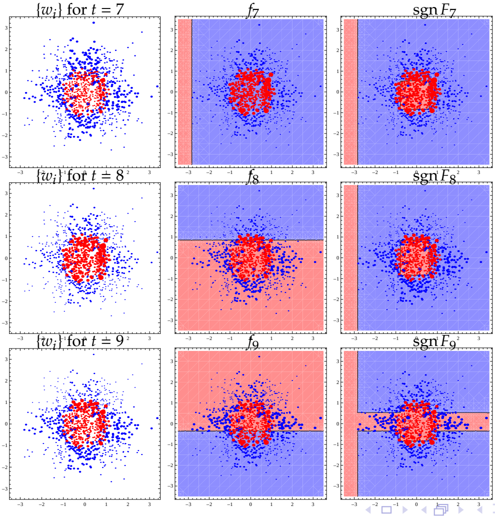
- Learning progress:



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

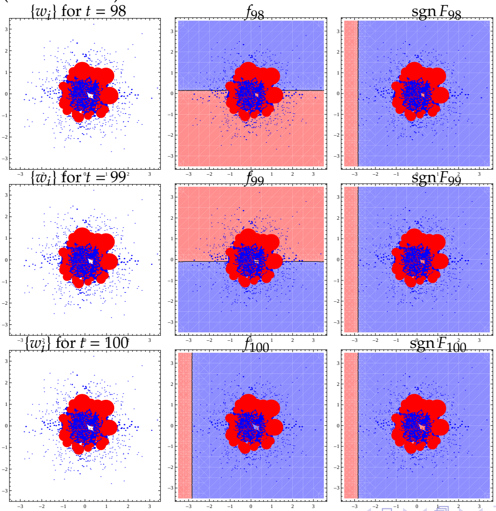
AdaBoost + decision stumps (5)

- Learning progress:



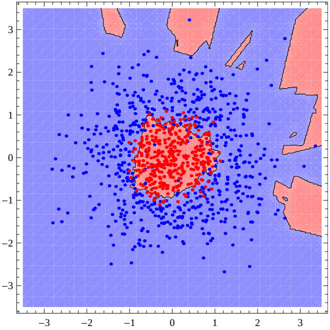
AdaBoost + decision stumps (6)

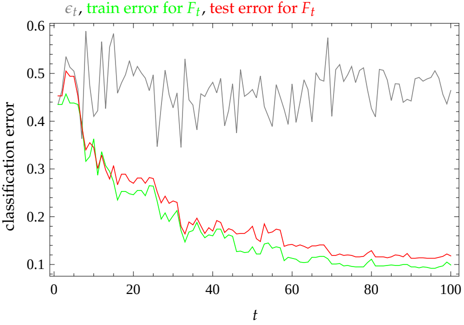
- Learning progress (last rounds):



AdaBoost + random lines (1)

- Weak classifiers: $f_t(\mathbf{x};\mathbf{c}) = 2[c_0 + c_1x_1 + c_2x_2 > 0] - 1$ with random coefficients drawn from $\mathbf{c} \in [-1,1]^3$.
- Final decision boundary for the ensemble ($T = 100$) and error plots:

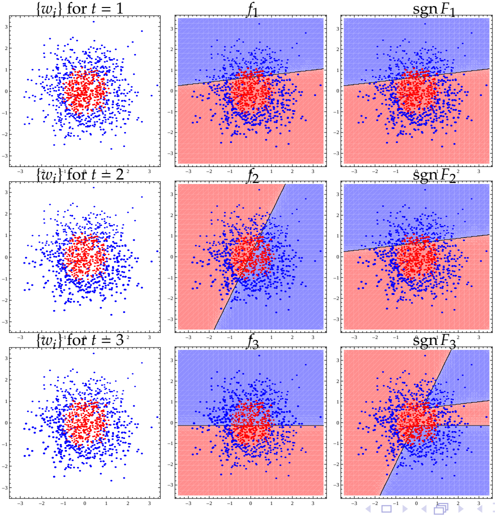




- True error: $\mathrm{err}_P(F) \approx 0.118755$.
- Error of F on test sample (also containing 1000 examples): 0.118.

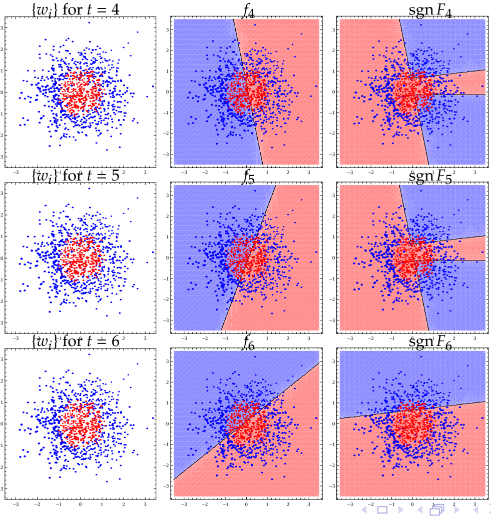
AdaBoost + random lines (2)

- Learning progress:



AdaBoost + random lines (3)

- Learning progress:



$\{w_i\}$ for $t = 4$

f_4

$\mathrm{sgn}\,F_4$

$\{w_i\}$ for $t = 5$

f_5

$\mathrm{sgn}\,F_5$

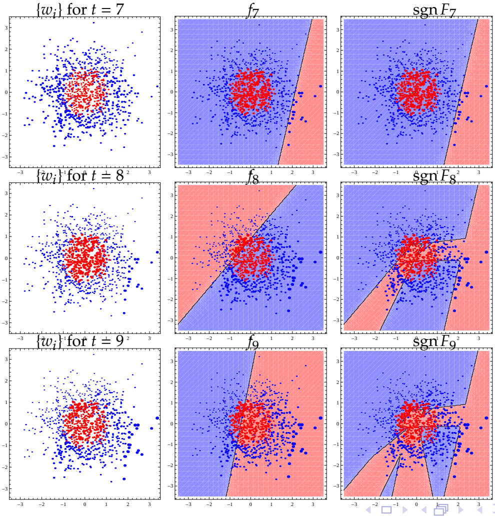
$\{w_i\}$ for $t = 6$

f_6

$\mathrm{sgn}\,F_6$

AdaBoost + random lines (4)

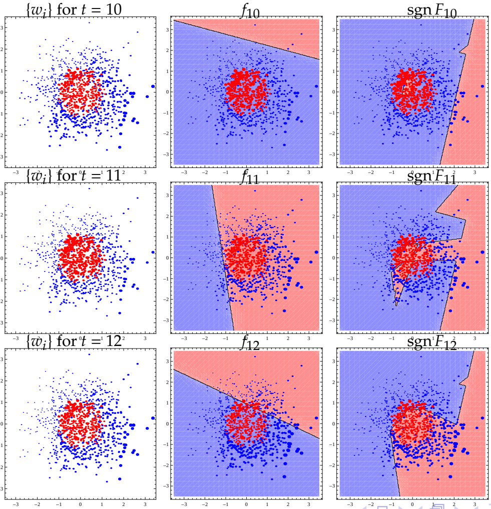
- Learning progress:



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

AdaBoost + random lines (5)

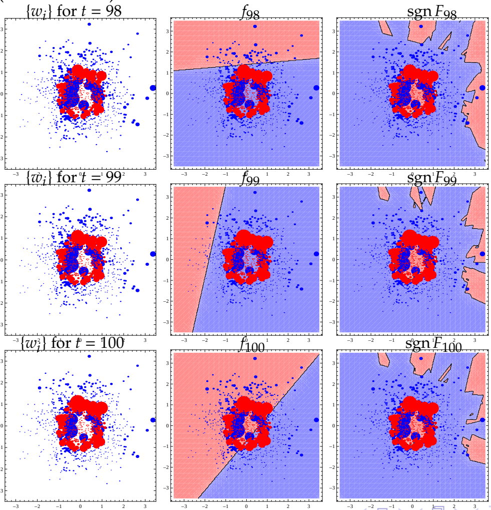
- Learning progress:



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

AdaBoost + random lines (6)

- Learning progress (last rounds):



AdaBoost — final remarks

- Popular variants:
  - AdaBoost + decision stumps,
  - AdaBoost + decision trees,
  - AdaBoost + linear classifiers (e.g. SVM),
  - AdaBoost + naive Bayes.
- “AdaBoost + decision stumps” variant is commonly synonymous with “Viola–Jones AdaBoost”.
- On a single boosting round the **choice of weak classifier** (step 4) can be performed using any error criterion (or even randomly).
- Typically though, two approches are most popular:
  (1) **classification error minimization**: $\arg\min_{f_t} \sum_{i=1}^{m} w_i [f_t(\mathbf{x}_i) \neq y_i]$,
  (2) **exponential creterion minimization**: $\arg\min_{f_t} \sum_{i=1}^{m} w_i e^{-\alpha_t f_t(\mathbf{x}_i) y_i}$.
- Optimal values for α_t are motivated by the exponential errors Z_t.
- If for some weak classifier $\epsilon_t > 1/2$ then the coefficient α_t shall **“negate” responses** to opposite ones.

AdaBoost — properties

Demonstrate that:

1. the choice $\alpha_t = \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t}$ minimizes the exponential criterion Z_t;
2. Z_t is equal to the ratio of exponential criterions on two consecutive rounds:

$$\sum_{i=1}^{m} e^{-y_i \sum_{j=1}^{t} \alpha_j f_j(\mathbf{x}_i)} \Bigg/ \sum_{i=1}^{m} e^{-y_i \sum_{j=1}^{t-1} \alpha_j f_j(\mathbf{x}_i)}; \tag{41}$$

3. train error for the ensemble F os upper-bounded by the product of Z_t values:

$$\frac{1}{m}\sum_{i=1}^{m}[\operatorname{sgn} F(\mathbf{x}_i) \neq y_i] \leqslant \prod_{t=1}^{T} Z_t; \tag{42}$$

4. . . . and thereby not greater than:

$$2^T \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)}. \tag{43}$$

RealBoost — initial remarks

- Idea in: *"Improved boosting using confidence-rated predictions"* **(Schapire & Singer, 1999)**.
- Full name: **Real AdaBoost** commonly shortened to **RealBoost**.
- Essence: **weak classifiers** are **real-valued** (not binary), i.e.. $f_t(\mathbf{x}) \in \mathbb{R}$.
- Response of a weak classifier is commonly set to approximate **half** the **logit** transform:

$$f_t(\mathbf{x}) = \frac{1}{2} \log \frac{\widehat{P}_w(y = 1|\mathbf{x})}{\widehat{P}_w(y = -1|\mathbf{x})}, \tag{44}$$

  where $\widehat{P}_w(y = \pm 1|\mathbf{x})$ estimates class distributions conditional on $\mathbf{x}$ using current weights w_i.
- E.g., for "decision stumps" when considering a classifier $f(\mathbf{x}; j, v, d)$ we have:

$$\widehat{P}_w(y = \pm 1|\mathbf{x}; j, v, d) = \begin{cases} \sum\limits_{\{i\colon d(x_{ij}-v)\leqslant 0,\ y_i=\pm 1\}} w_i, & \text{for } d(x_{ij} - v) \leqslant 0; \\ \sum\limits_{\{i\colon d(x_{ij}-v)>0,\ y_i=\pm 1\}} w_i, & \text{for } d(x_{ij} - v) > 0. \end{cases} \tag{45}$$

- In case of decision trees (applied as weak classifiers), each terminal produces its own estimation of $\widehat{P}_w(y = \pm 1|\mathbf{x})$.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

RealBoost — initial remarks

- **Ensemble classifier** is of form $F(\mathbf{x}) = \sum_{t=1}^{T} f_t(\mathbf{x})$ with decision: $\operatorname{sgn} F(\mathbf{x})$.
- One **resigns from coefficients of weak classifiers** — α_t — that were present in Discrete AdaBoost.
- Instead, a **weighing mechanism for classifiers** is built **in real-valued responses**.
- One can demonstrate that expression $1/2 \log\left(\widehat{P}_w(y=1|\mathbf{x}) \middle/ \widehat{P}_w(y=-1|\mathbf{x})\right)$ is the **solution** of **minimization of exponential criterion** defined by distribution $\{w_i\}$ on data (on a sample).
- Analogically, one can demonstrate that expression $1/2 \log\left(P(y=1|\mathbf{x}) \middle/ P(y=-1|\mathbf{x})\right)$ is a solution of minimization of exponential criterion defined by the **true uknown joint distribution** which generates data i.e. $P(\mathbf{x}, y) = p(\mathbf{x})P(y|\mathbf{x})$.
- One can observe similaries between **RealBoost** and **logistic regression**.

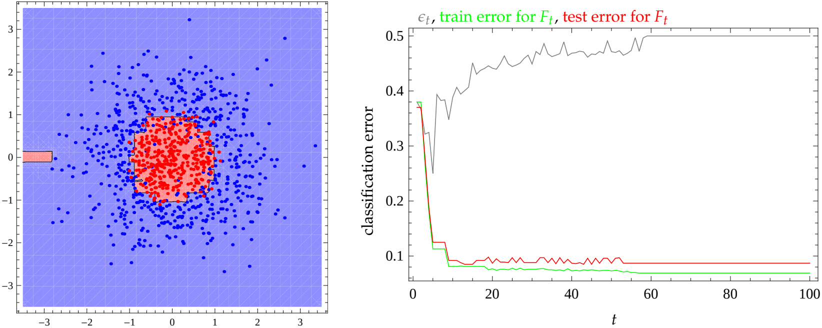
RealBoost

```
**algorithm** RealBoost($\mathcal{D}$)
    start with weights: $w_i := 1/m,\ i = 1, \ldots, m$.
    **for** $t := 1, \ldots, T$ **repeat**
        fit weak classifier $f_t(\mathbf{x}) \in \mathbb{R}$ using weights $w_i$ on training data, so that $f_t$
            minimizes *exponential criterion* $\sum_{i=1}^{m} w_i e^{-f_t(\mathbf{x}_i) y_i}$
            or equivalently so that $f_t$ approximates half the logit transform:
            $f_t(\mathbf{x}) := 1/2 \log \left( \widehat{P}_w(y = 1|\mathbf{x}) \big/ \widehat{P}_w(y = -1|\mathbf{x}) \right)$.
        update weights:
            $Z_t := \sum_{i=1}^{m} w_i e^{-f_t(\mathbf{x}_i) y_i}$.
            $w_i := w_i e^{-f_t(\mathbf{x}_i) y_i} / Z_t, \qquad i = 1, \ldots, m$.
    **return** ensemble $F(\mathbf{x}) := \sum_{t=1}^{T} f_t(\mathbf{x})$ with decision calculated as $\operatorname{sgn} F(\mathbf{x})$.
```

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

RealBoost + decision stumps

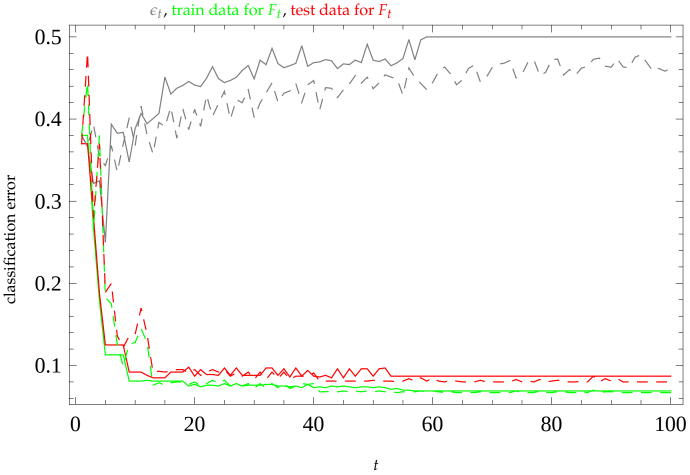
- Final decision boundary for the ensemble ($T = 100$) and error plots:



- True error: $\text{err}_P(F) \approx 0.092465$.
- Error of F on test sample (also containing 1000 examples): 0.087.

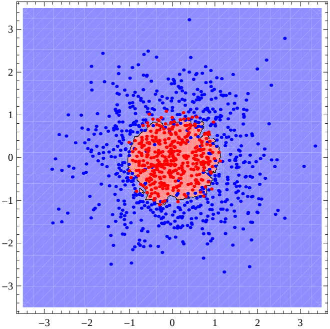
(RealBoost vs. AdaBoost) + decision stump

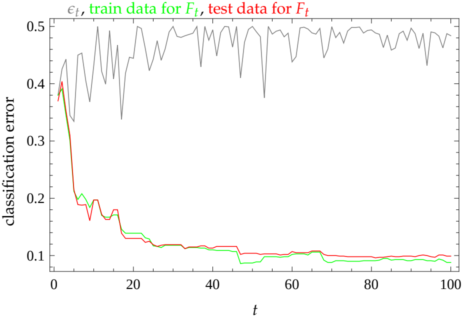
- Learning progress (*dashed curves for AdaBoost*):



(RealBoost vs. AdaBoost) + random lines

- Final decision boundary for the ensemble ($T = 100$) and error plots:

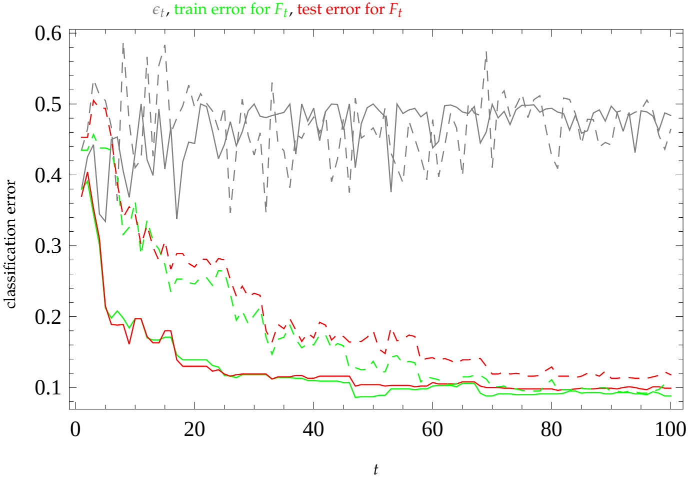




- True error: $\mathrm{err}_P(F) \approx 0.0975283$.
- Error of F on test sample (also containing 1000 examples): 0.099.

(RealBoost vs AdaBoost) + random lines

- Learning progress (*dashed curves for AdaBoost*):



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

RealBoost + normals

- Each weak classifier based on a single selected feature.
- Performed are **approximations of feature distributions conditional on classes** $p(x_j|y=\pm 1)$ via **normal distributions**: $\widehat{p}_w(x_j|y=\pm 1) = 1/\sqrt{2\pi\sigma^2_{j\pm}}e^{-(x_j-\mu_{j\pm})^2/(2\sigma^2_{j\pm})}$.
- Means and variances calculated as:

$$\mu_{j-} = \sum_{\{i:\, y_i=-1\}}^{m} w_i x_{ij} \Big/ \sum_{\{i:\, y_i=-1\}}^{m} w_i, \qquad \mu_{j+} = \sum_{\{i:\, y_i=1\}}^{m} w_i x_{ij} \Big/ \sum_{\{i:\, y_i=1\}}^{m} w_i, \tag{46}$$

$$\sigma^2_{j-} = \sum_{\{i:\, y_i=-1\}}^{m} w_i x^2_{ij} \Big/ \sum_{\{i:\, y_i=-1\}}^{m} w_i - \mu^2_{j-}, \qquad \sigma^2_{j+} = \sum_{\{i:\, y_i=-1\}}^{m} w_i x^2_{ij} \Big/ \sum_{\{i:\, y_i=1\}}^{m} w_i - \mu^2_{j+}. \tag{47}$$

- By virtue of Bayes theorem, response of a weak classifier becomes:

$$f_t(\mathbf{x}; j^*) = \frac{1}{2}\log\frac{\widehat{p}_w(x_{j^*}|y=1)\widehat{P}_w(y=1)}{\widehat{p}_w(x_{j^*}|y=-1)\widehat{P}_w(y=-1)} \tag{48}$$

$$= \frac{1}{2}\left(\frac{(x_{j^*}-\mu_{j^*-})^2}{2\sigma^2_{j^*-}} - \frac{(x_{j^*}-\mu_{j^*+})^2}{2\sigma^2_{j^*+}} + \log\frac{\sigma_{j^*-}}{\sigma_{j^*+}} + \log\frac{\widehat{P}_w(y=1)}{\widehat{P}_w(y=-1)}\right), \tag{49}$$

where $\widehat{P}_w(y=\pm 1)=\sum_{\{i:\, y_i=\pm 1\}} w_i$ and j^* indicates feature with smallest exponential criterion.

RealBoost + bins

- Idea in (Rasolzadeh et al., 2006) — similar to RealBoost + normals, but **conditional distributions approximated via piecewise constant functions** (of one variable) implemented using bins.
- Let $[a_1, a_2]$ represents interval of some feature, and B denotes the imposed number of bins (equally wide).
- Index of bin $\beta(x) \in \{1, \ldots, B\}$ that x belongs to is:

$$\beta(x) = \begin{cases} \lceil B(x-a_1)/(a_2-a_1) \rceil & \text{for } a_1 < x \leqslant a_2; \\ 1 & \text{for } x \leqslant a_1; \\ B & \text{for } a_2 < x. \end{cases} \tag{50}$$

- Let $\widehat{P}_w(y{=}-1, j \text{ inside bin } b) = \sum_{\{i\colon\, y_i=-1,\ \beta(x_{ij})=b\}} w_i$ denote estimated probability of the event that an example is negative and its j-th feature belonds to bin b.
- Response of a weak classifier (using j^*-th feature) is calculated as:

$$f_t(\mathbf{x}; j^*) = \frac{1}{2} \log \frac{\widehat{P}_w\left(y = 1, j^* \text{ inside bin } \beta(x_{j^*})\right)}{\widehat{P}_w\left(y = -1, j^* \text{ inside bin } \beta(x_{j^*})\right)}. \tag{51}$$

RealBoost + decision trees

- Idea based on well known **CART algorithm** (Breiman, Friedman, Olshen, & Stone, 1984).
- Practical experiments show that a set of shallow trees (obtained by boosting) typically performs better than a single deep tree.
- CART algorithm builds recursively a **binary tree** by splitting at each step a domain fragment via a **cut orthogonal to some axis (feature)**.
- Choice of the best split (j, v) — pair: (feature index, threshold) — is carried out by **minimizing expected impurity of children**.
- Popular impurities: *Gini index*, *entropy*.
- Tree terminals have real-valued responses equal to **halves** of the **logit transform**.
- Therefore, they are also **piecewise constant approximations** (similarly to RealBoost + bins) but of **several variables** (not univariate).

RealBoost + decision trees

- Consider a single step of recursion (for some tree node). Let $\{i\}$ denotes only those indexes of training examples that fall into the given node.
- For each split (j, v) we need the following quantities:

$$
\begin{aligned}
W(L) &= \sum_{\{i\colon x_{ij}<v\}} w_i, & W(y{=}-1, L) &= \sum_{\{i\colon x_{ij}<v,\ y_i=-1\}} w_i, & W(y{=}1, L) &= \sum_{\{i\colon x_{ij}<v,\ y_i=1\}} w_i,\\
W(R) &= \sum_{\{i\colon x_{ij}\geqslant v\}} w_i, & W(y{=}-1, R) &= \sum_{\{i\colon x_{ij}\geqslant v,\ y_i=-1\}} w_i, & W(y{=}1, R) &= \sum_{\{i\colon x_{ij}\geqslant v,\ y_i=1\}} w_i,
\end{aligned}
\tag{52}
$$

  where L and R denote left and right parts, respectively, resulting from the split.
- Probability estimates related to above quantities are:

$$
\begin{aligned}
\widehat{P}_w(L) &= W(L)/\left(W(L)+W(R)\right), & \widehat{P}_w(R) &= W(R)/\left(W(L)+W(R)\right),\\
\widehat{P}_w(y{=}-1|L) &= W(y{=}-1, L)/W(L), & \widehat{P}_w(y{=}1|L) &= W(y{=}1, L)/W(L),\\
\widehat{P}_w(y{=}-1|R) &= W(y{=}-1, R)/W(R), & \widehat{P}_w(y{=}1|R) &= W(y{=}1, R)/W(R).
\end{aligned}
\tag{53}
$$

- Expected impurity of children, e.g. for Gini index, becomes:

$$
\widehat{P}_w(L)\left(1-\widehat{P}^2_w(y{=}-1|L)-\widehat{P}^2_w(y{=}1|L)\right)+\widehat{P}_w(R)\left(1-\widehat{P}^2_w(y{=}-1|R)-\widehat{P}^2_w(y{=}1|R)\right).
\tag{54}
$$

- Each terminal returns: $1/2\log(\sum_{\{i\colon y_i=1\}} w_i / \sum_{\{i\colon y_i=-1\}} w_i)$.

RealBoost — properties

Demonstrate that:

1. expected value of exponential criterion:

$$\mathbb{E}_P\left(e^{-F(\mathbf{x})y}\right) = \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-F(\mathbf{x})y} P(y|\mathbf{x})p(\mathbf{x})\,\mathbf{dx} \tag{55}$$

   (with respect to true joint distribution P) attains its minimum for:

$$F(\mathbf{x}) = \frac{1}{2}\log\frac{P(y=1|\mathbf{x})}{P(y=-1|\mathbf{x})}; \tag{56}$$

2. by minimizing exponential criterion

$$Z_t = \sum_{i=1}^{m} w_{i,t} e^{-f_t(\mathbf{x}_i)y_i} \tag{57}$$

   in a *greedy* manner on each boosting round, one simultaneously minimizes that criterion for the ensemble, i.e.:

$$\frac{1}{m}\sum_{i=1}^{m} e^{-F(\mathbf{x}_i)y_i}. \tag{58}$$

Logistic regression

- A method for solving **classification task via linear regression approach**.
- We want to "model" the conditional distribution $P(y=1|\mathbf{x})$ by applying *somehow* a linear form $a_0 + a_1x_1 + \cdots + a_nx_n$.
- **Problem**: probabilities are bounded to $[0,1]$, whereas expression $a_0 + a_1x_1 + \cdots + a_nx_n$ is unbounded.
- **Trick**: instead of probabilities one can approximate **logarithmic odds ratio**:

$$\log\frac{P(y=1|\mathbf{x})}{1-P(y=1|\mathbf{x})}. \tag{59}$$

- By solving equation

$$a_0 + a_1x_1 + \cdots + a_nx_n = \log\frac{P(y=1|\mathbf{x})}{1-P(y=1|\mathbf{x})} \tag{60}$$

with respect to $P(y=1|\mathbf{x})$, one obtains **logistic function**:

$$P(y=1|\mathbf{x}) = \frac{1}{1+e^{-(a_0+a_1x_1+\cdots+a_nx_n)}} \tag{61}$$

(a.k.a. **sigmoid** function).

Logistic regression

- To solve logistic regression (to find $a_0, \ldots, a_n$) it is convenient to use $y_i \in \{0, 1\}$ instead of $y_i \in \{-1, 1\}$.
- To simplify notation denote $P(y = 1|\mathbf{x})$ as $p_i(\mathbf{x}_i)$.
- We build **likelihood function**:

$$L = \prod_{i=1}^{m} p(\mathbf{x}_i)^{y_i} \left(1 - p(\mathbf{x}_i)\right)^{1-y_i}. \tag{62}$$

- Its maximum with respect to $a_0, \ldots, a_n$ is in the same place as the maximum of **log-likelihood**:

$$\begin{aligned}\log L &= \sum_{i=1}^{m} \Big(y_i \log p(\mathbf{x}_i) + (1 - y_i) \log\left(1 - p(\mathbf{x}_i)\right) \Big) \\ &\vdots \\ &= \sum_{i=1}^{m} \Big(y_i (a_0 + a_1 x_{i1} + \cdots a_n x_{in}) - \log\left(1 + e^{a_0 + a_1 x_{i1} + \cdots a_n x_{in}}\right) \Big). \end{aligned} \tag{63}$$

Connection: RealBoost ~ logistic regression

- Consider **expectation of exponential criterion** taken with respect to true distribution P, from which pairs $(\mathbf{x}, y)$ are drawn:

$$Q_P(F) = \mathbb{E}_P\Big(e^{-yF(\mathbf{x})}\Big) = \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-yF(\mathbf{x})} p(\mathbf{x}, y)\,\mathbf{dx}$$
$$= \int_{\mathbf{x}} \Big(P(y{=}-1|\mathbf{x})e^{F(\mathbf{x})} + P(y{=}1|\mathbf{x})e^{-F(\mathbf{x})}\Big) p(\mathbf{x})\,\mathbf{dx}. \qquad (64)$$

- We know that by demanding $\partial Q_P(F)/\partial F = 0$ one obtains solution:

$$F^*(\mathbf{x}) = 1/2 \log\Big(P(y{=}1|\mathbf{x})/P(y{=}-1|\mathbf{x})\Big) \qquad (65)$$

  (in fact, it suffices to minimize the inner expectation in (64) for conditional distribution $P(y = \pm 1|\mathbf{x})$).
- Note that F^* is **half** the **logit** transform, typical for logistic regression.
- If the learning algorithm was capable *somehow* of finding immediately (in one step) the optimal function F^* then the **boosting procedure could be stopped after just one round**.
- In practice, **weak classifiers are crude approximations** of F^*, therefore multiple rounds are needed.

Connection: RealBoost ~ logistic regression

- Solving (65) with respect to $P(y = 1|\mathbf{x})$ one obtains a form of **sigmoid**:

$$P(y = 1|\mathbf{x}) = e^{2F^*(\mathbf{x})} \Big/ \left(1 + e^{2F^*(\mathbf{x})}\right) = 1 \Big/ \left(1 + e^{-2F^*(\mathbf{x})}\right), \tag{66}$$

  — equivalent to logistic regression up to a constant factor of 2 in the exponent.

- **Logistic regression approximates** F^* by a **linear model**:

$$F^*(\mathbf{x}) \approx a_0 + a_1 x_1 + \cdots + a_n x_n. \tag{67}$$

- **RealBoost approximates** F^* by a **linear combination of weak classifiers**:

$$F^*(\mathbf{x}) \approx f_1(\mathbf{x}) + \cdots + f_T(\mathbf{x}), \tag{68}$$

  therefore by simple functions but possibly of multiple variables each.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Connection: RealBoost ∼ error residuals

- Consider the technique of **error residuals** known from regression.
- Using it, we **sequentially build an additive model**, where each successive fragment of approximation *"explains"* some part of the target quantity and becomes subtracted from it, so that the **next fragments concentrate on error residuals**.
- **Reweighing scheme in boosting works analogically to error residuals**.

Connection: RealBoost ~ error residuals

- Suppose we have a partial model F and we want to update it to $F := F + f$.
- Consider a **population-based** version of **boosting** (aware of distribution P).
- For reweighing formulas based on data examples: $Z = \frac{1}{m}\sum_{i=1}^{m} e^{-y_i F(\mathbf{x}_i)}$, $w_i = e^{-y_i F(\mathbf{x}_i)}/Z$, we can define their population-based counterparts pertaining to P:

$$Z = \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-yF(\mathbf{x})} p(\mathbf{x},y)\,\mathbf{dx}; \qquad w(\mathbf{x},y) = p(\mathbf{x},y)e^{-yF(\mathbf{x})}/Z. \tag{69}$$

- Z works as a normalizing constant, but simultaneously $Z = Q_P(F)$ — optimization criterion value for the model obtained so far.
- Consider the value of criterion for $F + f$:

$$\begin{aligned} Q_P(F{+}f) &= \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-y\left(F(\mathbf{x})+f(\mathbf{x})\right)} p(\mathbf{x},y)\,\mathbf{dx} \\ &= \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-yf(\mathbf{x})} \underbrace{e^{-yF(\mathbf{x})} p(\mathbf{x},y)/Z}_{w(\mathbf{x},y)}\,\mathbf{dx}\cdot Z {=} Q_w(f)\cdot Q_P(F). \end{aligned} \tag{70}$$

- **Conclusion: to minimize $Q_P(F+f)$ it suffices to greedily minimize $Q_w(f)$; the current state of distribution $w(\mathbf{x},y)$ indicates which places of target quantity are already approximated well (“explained”) and which places still require approximation.**

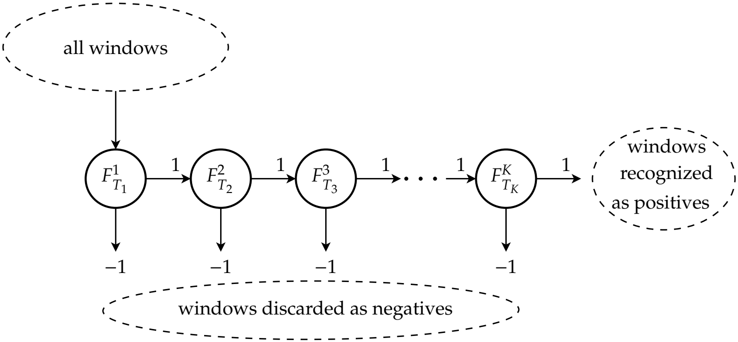
Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

Sketch of idea

- Observation: **negative windows are vast majority of all windows** (commonly $\geqslant 99.9\%$).
- Therefore, it is worth to build **simpler classifiers using fewer features** serving to **discard negative windows faster**.
- Windows that appear promising (for positives) can be analyzed longer, using more features.
- A sequence of classifiers gets trained: $F^1_{T_1}, F^2_{T_2}, \ldots$, forming a **cascade** — **binary tree degenerated to a list**.
- Sizes of successive classifiers in the cascaded form a non-decreasing sequence: $T_1 \leqslant T_2 \leqslant \ldots$.
- Names for successive elements: **stages**, levels, layers.
- **Positive** indication requires traversing **all stages**.
- **Negative** indication on **any stage stops** further analysis.

Scheme of cascade



- E.g. in (Viola & Jones, 2004): $K = 32$ stages with $T_1 = 2$, $T_2 = 5$, $T_3 = \cdots = T_5 = 20$, $T_6 = T_7 = 50$, $T_8 = \cdots = T_{12} = 100$, $T_{13} = \cdots = T_{32} = 200$.
- In total: 4 297 features, **on average ≈ 8 features extracted per window**.

Final requirements and stage requirements

- When training, one has to adjust **decision thresholds** θ_k which influence stage decisions: $\text{sgn}\left(F^k_{T_k}(\mathbf{x}) - \theta_k\right)$, so that **each stage** has **very high sensitivity** (a.k.a. detection rate), for example $\geqslant 99.9\%$, and **moderately small false alarm rate** (1− specifity), for example $< 50\%$.
- Let $d_1, d_2, \ldots, d_K$ denote a sequence of sensitivities for successive cascade stages, and $a_1, a_2, \ldots, a_K$ the corresponding sequence of FAR values.
- **Final sensitivity and FAR** for the whole cascade are equal to:

$$D = \prod_{k=1}^{K} d_k, \tag{71}$$

$$A = \prod_{k=1}^{K} a_k. \tag{72}$$

- Having imposed **requirements for whole casced** (i.e.. D, A), one can derive partial requirements: $d_{\min}$ and $a_{\max}$ for each stage.
- E.g. for $D = 0.98$, $A = 10^{-5}$ and $K = 10$, it suffices that each stage satisfies $d_i \geqslant d_{\min} = 0.998$ (because $0.998^{10} \geqslant 0.98$) and $a_i \leqslant a_{\max} = 0.316$ (because $0.316^{10} \leqslant 10^{-5}$).

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Cascade learning algorithm — remarks

- User imposes wanted stage requirements: $d_{\min}$, $a_{\max}$.
- Each stage is trained by a boosting algorithm (e.g. AdaBoost or RealBoost).
- The number of classifiers for given stage is increased one-by-one until the stage satisfies $d_{\min}$, $a_{\max}$.
- Quantities d_i, a_i (sensitivity, FAR) obtained at current stage are measured on a separate **validation set**.
- After each weak classifier is added, the decision threshold θ_k becomes updated (typically lowered), so that the wanted sensitivity is met: $d_k \geqslant d_{\min}$. In consequence, this also increases the observed FAR i.e. a_k.
- The cascade is extended with new stages until the overall requirements D, A are met.

Remark 1: It is not clear if decision threshold updates do not worsen generalization capabilities.
Remark 2: Stop condition might not be reached.

Cascade learning algorithm

```
algorithm TrainCascade($\mathcal{D}, D, A, d_{\min}, a_{\max}, \mathcal{V}$)    ▷ $\mathcal{V}$ — validating set
    distinguish positive and negative subsets in data: $\mathcal{P}$, $\mathcal{N}$ within $\mathcal{D}$
    $D_0 := 1, A_0 := 1, k := 0$.
    while $A_k > A$ repeat    ▷ $A_k$ — overall FAR for $k$ initial stages
        $k := k + 1, T_k := 0, A_k := A_{k-1}, F^k := 0$.    ▷ subindexes $F^k_{T_k}$ skipped
        while $A_k > a_{\max} \cdot A_{k-1}$ repeat
            $T_k := T_k + 1$
            use $\mathcal{P}$ and $\mathcal{N}$ to fit new weak classifier $f$, obtaining $F^k := F^k + f$
            update $\theta_k$ for ensemble $F^k$, so that sensitivity for whole cascade is
                $D_k \geqslant d_{\min} \cdot D_{k-1}$, as follows: $\theta_k := F^k(\mathcal{V}_+)_{\lfloor(1-d_{\min})\cdot\#\mathcal{V}_+\rfloor}$,
                where $F^k(\mathcal{V}_+)$ denotes sorted sequence of real-valued responses
                of $F^k$ with respect to positive examples among $\mathcal{V}$    ▷ by that we also increase $A_k$
            execute current cascade $(F^1, F^2, \ldots, F^k)$ on $\mathcal{V}$ to measure its $D_k$ and $A_k$
        if $A_k > A$ then
            $\mathcal{N} := \emptyset$.
            execute current cascade $(F^1, F^2, \ldots, F^k)$ on newly sampled windows
                from negative images and add false alarms to $\mathcal{N}$
        otherwise
            break
    return cacscade $(F^1, F^2, \ldots, F^k)$.
```

Table of contents

1. Object detection — overview
2. Orthogonal expansions
3. Haar wavelets and Haar-like features
4. Practical example: face detector (HFs)
5. Practical example: landmine detector (3D HFs)
6. HOG descriptor
7. Boosting — variants and properties
8. Cascades of classifiers
9. Towards new results

New repertoire of features / integral images

1. **Fast Fourier moments (Klęsk, 2017):**

$$ii^{\cos}_{t,u}(x,y)=\sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} i(j,k)\cos(\cdots;j,k,t,u),\quad ii^{\sin}_{t,u}(x,y)=\sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} i(j,k)\sin(\cdots;j,k,t,u).\tag{73}$$

(constant-time features; ≈ 2 times slower than HFs; more accurate than HFs in face detection)

2. **Fast statistical moments (Klęsk & Bera, 2018):**

$$ii_{t,u}(x,y)=\sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} i(j,k)j^t k^u.\tag{74}$$

(constant-time features; ≈ 3 times slower than HFs)

3. **Fast Zernike moments (Bera, Klęsk, & Sychel, 2018):**

$$ii_{t,u}(x,y)=\sum_{1\leqslant j\leqslant x}\sum_{1\leqslant k\leqslant y} i(j,k)(k-ij)^t(k+ij)^u,\quad \text{where } i^2=-1.\tag{75}$$

(constant-time features; ≈ 12 times slower than HFs; rotationally invariant)

In all cases, computations faster than definition-style computations by $> 10^3$ times.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

FMs backed with integral images

- A technique for **constant-time** calculation of **low order Fourier moments**, applicable in detection tasks.
- Real and imaginary parts of moments can be used as features.
- Technique based on a ***set*** of special ***integral images*** involving ***trigonometric terms***.
- Additional time invested in integral images amortized during detection.
- Extraction of each feature requires **21 operations**, regardless of detection window size and position — ***O*(1) calculation**.
- Experiments on **face detection**: Fourier moments vs. Haar-like features.

Fourier moments

- Consider the following **approximation**, by a **partial Fourier sum**, of an image fragment restricted to a rectangle spanning from (x_1, y_1) to (x_2, y_2):

$$i(x,y) \approx \sum_{-n\leqslant k_x\leqslant n} \sum_{-n\leqslant k_y\leqslant n} c^{k_x,k_y}_{\substack{x_1,y_1\\x_2,y_2}} e^{2\pi i\left(k_x \frac{x-x_1}{N_x} + k_y \frac{y-y_1}{N_y}\right)}, \qquad \substack{x_1\leqslant x\leqslant x_2;\\ y_1\leqslant y\leqslant y_2,} \tag{76}$$

  where: n — harmonic order of approximation, $i = \sqrt{-1}$ — imaginary unit, c — complex coefficients, and $N_x{=}x_2{-}x_1{+}1$, $N_y{=}y_2{-}y_1{+}1$ — rectangle widths in pixels.

- Best coefficients — the **moments** are:

$$c^{k_x,k_y}_{\substack{x_1,y_1\\x_2,y_2}} = \frac{1}{N_x N_y} \sum_{x_1\leqslant x\leqslant x_2} \sum_{y_1\leqslant y\leqslant y_2} i(x,y) e^{-2\pi i\left(k_x \frac{x-x_1}{N_x} + k_y \frac{y-y_1}{N_y}\right)}. \tag{77}$$

Proposition

- We introduce ***two sets of integral images***:

$$\left\{ii_{\cos\atop N_x,N_y}^{k_x,k_y}\right\},\quad \left\{ii_{\sin\atop N_x,N_y}^{k_x,k_y}\right\},$$

constructed as:

$$ii_{\cos\atop N_x,N_y}^{k_x,k_y}(x,y)=\sum_{1\leqslant j_x\leqslant x}\sum_{1\leqslant j_y\leqslant y} i(j_x,j_y)\cos\left(-2\pi\left(\frac{k_xj_x}{N_x}+\frac{k_yj_y}{N_y}\right)\right), \tag{78}$$

$$ii_{\sin\atop N_x,N_y}^{k_x,k_y}(x,y)=\sum_{1\leqslant j_x\leqslant x}\sum_{1\leqslant j_y\leqslant y} i(j_x,j_y)\sin\left(-2\pi\left(\frac{k_xj_x}{N_x}+\frac{k_yj_y}{N_y}\right)\right), \tag{79}$$

where indexes (k_x,k_y) iterate over

$$\Big\{(k_x,k_y)\colon\ -n\leqslant k_x\leqslant -1, -n\leqslant k_y\leqslant n\Big\}\cup\Big\{(0,k_y)\colon\ -n\leqslant k_y\leqslant -1\Big\}\cup\{(0,0)\}. \tag{80}$$

- Each integral image can be calculated in linear time with respect to image size (induction).
- Define the ***growth operator*** for any integral image from $\{ii_{\cos}\}$ or $\{ii_{\sin}\}$:

$$\Delta_{x_1,y_1\atop x_2,y_2}(ii)=ii(x_2,y_2)-ii(x_1-1,y_2)-ii(x_2,y_1-1)+ii(x_1-1,y_1-1). \tag{81}$$

Proposition

Proposition 1

Suppose the two sets of integral images $\left\{ii^{k_x,k_y}_{\substack{\cos\\ N_x,N_y}}\right\}$, $\left\{ii^{k_x,k_y}_{\substack{\sin\\ N_x,N_y}}\right\}$*, defined as in* (78) *and* (79)*, have been calculated prior to the detection procedure. Then, for any rectangle of widths* N_x, N_y *in the image, the real and imaginary parts of each of its Fourier moments can be calculated in constant time —* $O(1)$ *— as follows:*

$$Re\left(c^{k_x,k_y}_{\substack{x_1,y_1\\ x_2,y_2}}\right) = \frac{1}{N_x N_y}\left(\cos\left(2\pi\left(\frac{k_x x_1}{N_x}+\frac{k_y y_1}{N_y}\right)\right)\Delta_{\substack{x_1,y_1\\ x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\cos\\ N_x,N_y}}\right) - \sin\left(2\pi\left(\frac{k_x x_1}{N_x}+\frac{k_y y_1}{N_y}\right)\right)\Delta_{\substack{x_1,y_1\\ x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\sin\\ N_x,N_y}}\right)\right), \tag{82}$$

$$Im\left(c^{k_x,k_y}_{\substack{x_1,y_1\\ x_2,y_2}}\right) = \frac{1}{N_x N_y}\left(\sin\left(2\pi\left(\frac{k_x x_1}{N_x}+\frac{k_y y_1}{N_y}\right)\right)\Delta_{\substack{x_1,y_1\\ x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\cos\\ N_x,N_y}}\right) + \cos\left(2\pi\left(\frac{k_x x_1}{N_x}+\frac{k_y y_1}{N_y}\right)\right)\Delta_{\substack{x_1,y_1\\ x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\sin\\ N_x,N_y}}\right)\right). \tag{83}$$

[in total 21 operations: 8 additions/subtractions, 8 multiplications, 3 divisions, and 2 trigonometric]

Proposition

Proof: Rewrite the moments using Euler's identity —

$$c^{k_x,k_y}_{\substack{x_1,y_1\\x_2,y_2}} = \frac{1}{N_x N_y}\sum_{x_1\leqslant x\leqslant x_2}\sum_{y_1\leqslant y\leqslant y_2} i(x,y)\left(\cos\left(-2\pi\left(k_x\frac{x-x_1}{N_x}+k_y\frac{y-y_1}{N_y}\right)\right)+i\sin\left(-2\pi\left(k_x\frac{x-x_1}{N_x}+k_y\frac{y-y_1}{N_y}\right)\right)\right). \tag{84}$$

Part the argument of the trigonometric functions into a group of terms independent from the pixel index (x,y) and a group dependent on it as follows:

$$\alpha = 2\pi\left(k_x x_1/N_x + k_y y_1/N_y\right), \qquad \beta(x,y) = -2\pi\left(k_x x/N_x + k_y y/N_y\right).$$

Apply in (84) the trigonometric identities for $\cos(\alpha+\beta)$ and $\sin(\alpha+\beta)$. Simultaneously, pull terms $\cos\alpha$ and $\sin\alpha$ in front of summations — independent of the pixel index (x,y). Finally, split the expression into real and imaginary parts:

$$\mathrm{Re}\left(c^{k_x,k_y}_{\substack{x_1,y_1\\x_2,y_2}}\right)=\frac{1}{N_xN_y}\left(\cos\alpha\underbrace{\sum_{\substack{x_1\leqslant x\leqslant x_2\\y_1\leqslant y\leqslant y_2}} i(x,y)\cos\beta(x,y)}_{\Delta_{\substack{x_1,y_1\\x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\cos\\N_x,N_y}}\right)}-\sin\alpha\underbrace{\sum_{\substack{x_1\leqslant x\leqslant x_2\\y_1\leqslant y\leqslant y_2}} i(x,y)\sin\beta(x,y)}_{\Delta_{\substack{x_1,y_1\\x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\sin\\N_x,N_y}}\right)}\right),$$

$$\mathrm{Im}\left(c^{k_x,k_y}_{\substack{x_1,y_1\\x_2,y_2}}\right)=\frac{1}{N_xN_y}\left(\sin\alpha\underbrace{\sum_{\substack{x_1\leqslant x\leqslant x_2\\y_1\leqslant y\leqslant y_2}} i(x,y)\cos\beta(x,y)}_{\Delta_{\substack{x_1,y_1\\x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\cos\\N_x,N_y}}\right)}+\cos\alpha\underbrace{\sum_{\substack{x_1\leqslant x\leqslant x_2\\y_1\leqslant y\leqslant y_2}} i(x,y)\sin\beta(x,y)}_{\Delta_{\substack{x_1,y_1\\x_2,y_2}}\left(ii^{k_x,k_y}_{\substack{\sin\\N_x,N_y}}\right)}\right).$$

Underbraces show how the expensive summations over pixels get replaced by constant-time growths of integral images. ■

Producing more features

- Constant-time extraction owed to additional costs invested in integral images.
- Needed **number of integral images**: $(2n+1)^2+1$, since both $ii_{\cos}, ii_{\sin}$ are required for each k_x, k_y pair — potentially expensive, hence **low harmonic orders** (e.g. $n = 1, 2, 3$).
- We **partition windows** into regular $p \times p$ grids of rectangles:

original

96×96

$n = 4$ ($p = 1$)

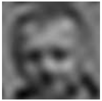
features: 81

feats/pxs: 0.0088

MAE: 0.0711

$n = 5$ ($p = 1$)

features: 121

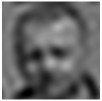


feats/pxs: 0.0131

MAE: 0.0600

$n = 6$ ($p = 1$)

features: 169

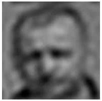


feats/pxs: 0.0183

MAE: 0.0554

$n = 7$ ($p = 1$)

features: 225



feats/pxs: 0.0244

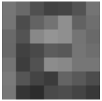
MAE: 0.0506

original

96×96

$n = 0$ ($p = 7$)

features: 49

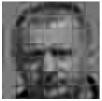


feats/pxs: 0.0059

MAE: 0.0833

$n = 1$ ($p = 7$)

features: 441

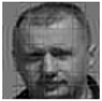


feats/pxs: 0.0533

MAE: 0.0496

$n = 2$ ($p = 7$)

features: 1 225

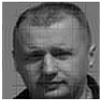


feats/pxs: 0.1479

MAE: 0.0377

$n = 3$ ($p = 7$)

features: 2 401



feats/pxs: 0.2899

MAE: 0.0305

- Features extracted from each rectangle — **piecewise Fourier approximation**.
- **Total number of features:** $d(n,p) = (2n+1)^2 p^2$.

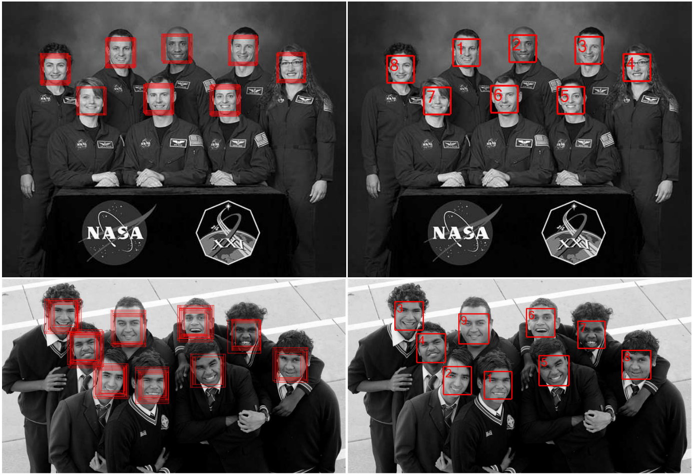
Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Experimental setup

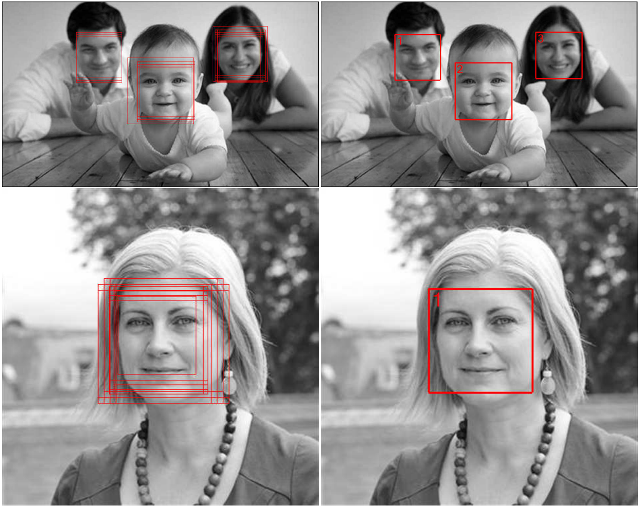
- Four variants of **Fourier-based face detectors**:
  (1) $n = 2, p = 5$ (625 feats.),
  (2) $n = 2, p = 7$ (1 225 feats.),
  (3) $n = 3, p = 5$ (1 225 feats.),
  (4) $n = 3, p = 7$ (2 401 feats.)
- **Haar-like opponents** (5 templates, q scales per axis, same grid sizes):
  (1) $q = 3, p = 5$ (1 125 feats.),
  (2) $q = 3, p = 7$ (2 000 feats.),
  (3) $q = 4, p = 5$ (2 205 feats.),
  (4) $q = 4, p = 7$ (3 920 feats.)
- **Train data**: 7 258 positive examples, 100 000 negative examples.
- **Learning algorithm**: **RealBoost + bins**, ensemble sizes: $T = 256$ or $T = 512$.
- **Test data**: 70 252 859 windows within 500 images containing 1 000 faces.
- To conveniently generate **ROCs** a test *subset* generated with $2 \cdot 10^6$ negatives → precision along FAR axis: $5 \cdot 10^{-7}$.
- **Detection procedure 1 (“heavy”):** ≈ 151 000 windows (8 scales, sliding window 48×48 up to 172×172, jumps ratio 0.05).
- **Detection procedure 2 (“light”):** ≈ 11 000 windows (4 scales, sliding window 120×120 up to 207×207, jumps ratio 0.05)
- Software written in **C#** with key procedures in **C++** as dll libraries.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

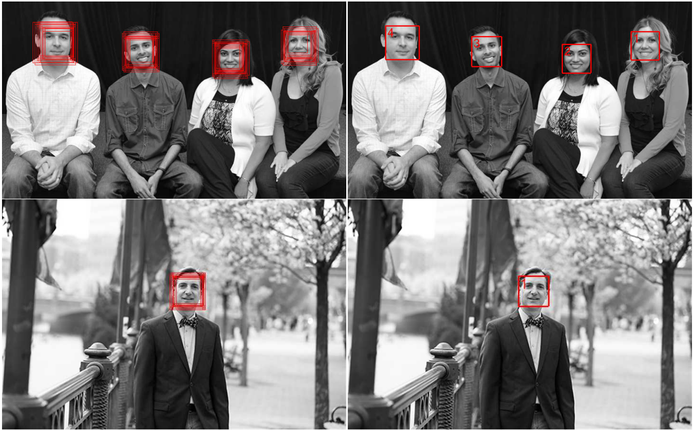
Examples of outcomes (Fourier moments)



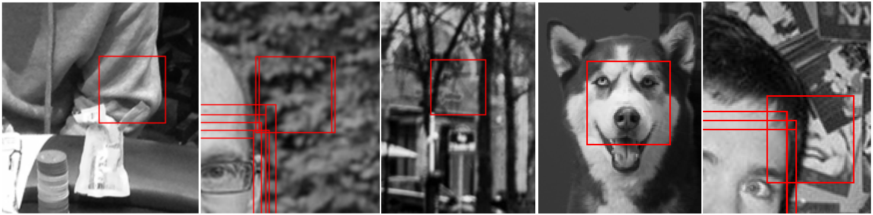
Examples of outcomes (Fourier moments)



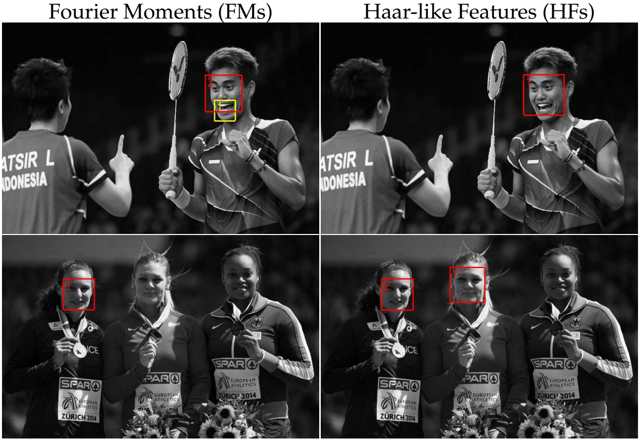
Examples of outcomes (Fourier moments)



Examples of false alarms

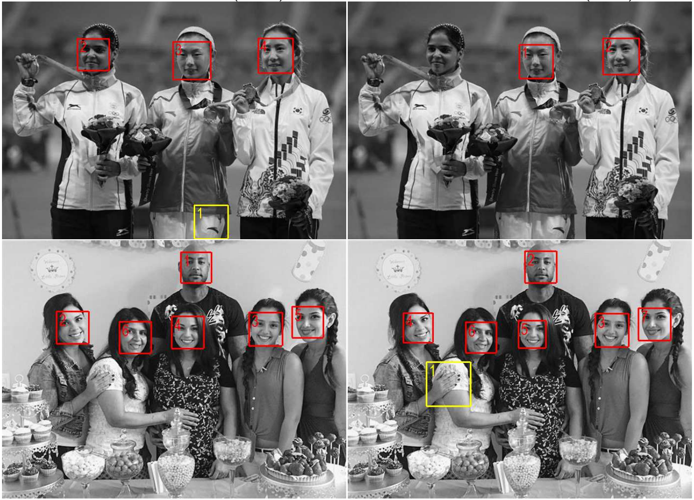


Some examples FMs vs. HFs (errors)



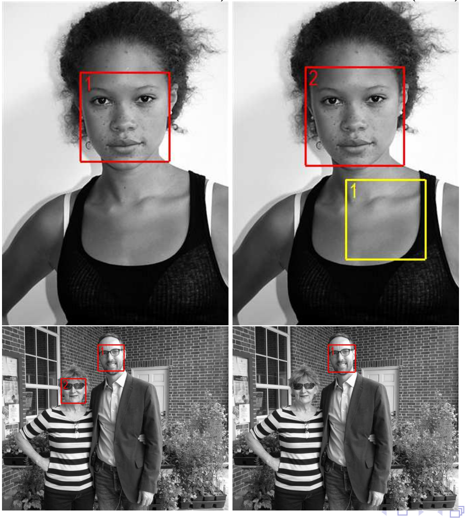
Some examples FMs vs. HFs (errors)

Fourier Moments (FMs) | Haar-like Features (HFs)



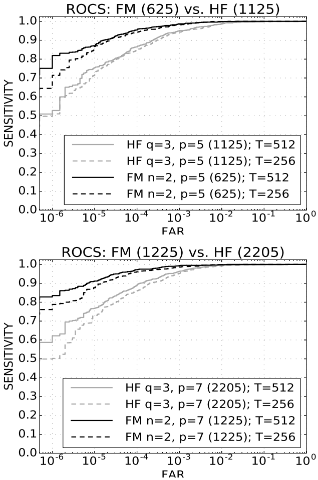
Some examples FMs vs. HFs (errors)

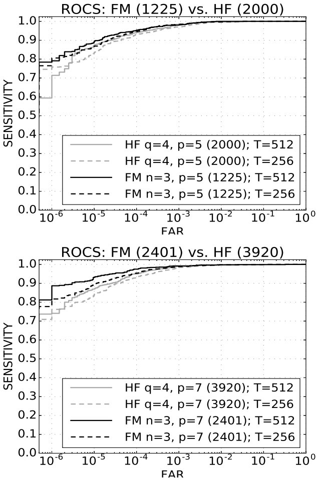
Fourier Moments (FMs) Haar-like Features (HFs)



Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

ROC curves


ROCS: FM (625) vs. HF (1125)
HF q=3, p=5 (1125); T=512
HF q=3, p=5 (1125); T=256
FM n=2, p=5 (625); T=512
FM n=2, p=5 (625); T=256
ROCS: FM (1225) vs. HF (2205)
HF q=3, p=7 (2205); T=512
HF q=3, p=7 (2205); T=256
FM n=2, p=7 (1225); T=512
FM n=2, p=7 (1225); T=256
SENSITIVITY
FAR
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}


ROCS: FM (1225) vs. HF (2000)
HF q=4, p=5 (2000); T=512
HF q=4, p=5 (2000); T=256
FM n=3, p=5 (1225); T=512
FM n=3, p=5 (1225); T=256
ROCS: FM (2401) vs. HF (3920)
HF q=4, p=7 (3920); T=512
HF q=4, p=7 (3920); T=256
FM n=3, p=7 (2401); T=512
FM n=3, p=7 (2401); T=256
SENSITIVITY
FAR
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}

Accuracy measures

name / description	AUC_α			sensitivity	FAR per image	FAR per window	accuracy per window
	α=10^{-5}	α=10^{-4}	α=10^{-3}				
HF q=3, p=5 (1125); T=512	0.6761	0.8123	0.9156	0.699	0.098	$6.975{\cdot}10^{-7}$	0.999995018084708
FM n=2, p=5 (625); T=512	0.8401	0.9236	0.97144	0.889	0.092	$6.548{\cdot}10^{-7}$	0.999997765249097
HF q=4, p=5 (2000); T=512	0.8021	0.9082	0.9624	0.849	0.086	$6.121{\cdot}10^{-7}$	0.999997238589628
FM n=3, p=5 (1225); T=512	0.8475	0.9285	0.9703	0.872	0.054	$3.843{\cdot}10^{-7}$	0.999997793717795
HF q=3, p=7 (2205); T=512	0.7075	0.8376	0.9320	0.741	0.084	$5.978{\cdot}10^{-7}$	0.999995715550288
FM n=2, p=7 (1225); T=512	0.8800	0.9480	0.9826	0.924	0.058	$4.128{\cdot}10^{-7}$	0.999998505420626
HF q=4, p=7 (3920); T=512	0.8188	0.9141	0.9729	0.897	0.102	$7.234{\cdot}10^{-7}$	0.999997815602899
FM n=3, p=7 (2401); T=512	0.8965	0.9538	0.9845	0.951	0.062	$4.397{\cdot}10^{-7}$	0.999998865248259

Time performance — “heavy” procedure

	Fourier moments ($T = 512$)	Haar-like features ($T = 512$)
quantity (or operations)	(456 distinct feats.)	(472 distinct feats.)
no. of analyzed windows	151 385	151 385
no. of prepared integral images	400 (50 images per each of 8 scales)	1
preparation time for integral images	421 ms	6 ms
preparation time per 1 integral image	1.05 ms	6 ms
total time of detection procedure	1 190 ms	513 ms
time per 1 window	7.86 μs (amortized: 5.08 μs)	3.38 μs (amortized: 3.34 μs)
time per 1 window and 1 feature	17.59 ns (amortized: 11.36 ns)	6.83 ns (amortized: 6.75 ns)

[640 × 480 image; parallel computations on: Intel Xeon E3-1505M v5 4×2-core 2.80 (3.70) GHz CPU;]

[cascade of classifiers *not* used]

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Time performance — “light” procedure

quantity (or operations)	Fourier moments ($T = 512$) (456 distinct feats.)	Haar-like features ($T = 512$) (472 distinct feats.)
no. of analyzed windows	11 838	11 838
no. of prepared integral images	200 (50 images per each of 4 scales)	1
preparation time for integral images	219 ms	6 ms
preparation time per 1 integral image	1.10 ms	6 ms
total time of detection procedure	308 ms	83 ms
time per 1 window	26.02 μs (amortized: 7.52 μs)	7.01 μs (amortized: 6.50 μs)
time per 1 window and 1 feature	58.21 ns (amortized: 16.82 ns)	14.16 ns (amortized: 13.14 ns)

[640 × 480 image; parallel computations on: Intel Xeon E3-1505M v5 4×2-core 2.80 (3.70) GHz CPU;]

[cascade of classifiers *not* used]

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

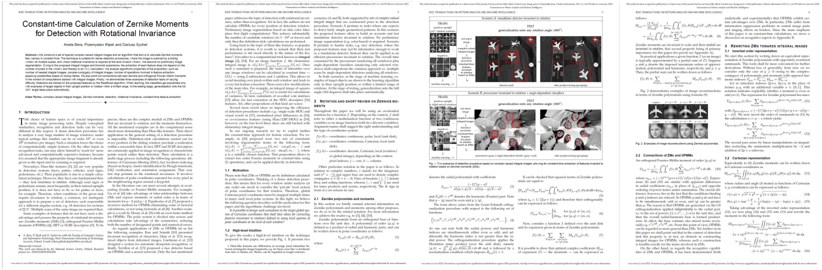
FMs — conclusions

1. **Algorithmic result for detection tasks:** a computational technique for **constant-time extraction of low order Fourier moments**, based on **special integral images**.
2. Experiments have shown that **fairly small sets of Fourier-based features can surpass Haar-like features in terms of accuracy in face detection task**.
3. Approach can be beneficial in machine learning applications where **accuracy is of primary importance rather than real-time** (e.g.: medical diagnosis, image-based fault detection, landmine detection) .
4. Real-time could be achieved with: cascade of classifiers + more parallelism (e.g. more CPU cores) to prepare integral images.

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

Paper on constant-time Zernike Moments

- **A. Bera, P. Klęsk, and D. Sychel: "Constant-time Calculation of Zernike Moments for Detection with Rotational Invariance",**
  IEEE Transactions on Pattern Analysis and Machine Intelligence,
  DOI: 10.1109/TPAMI.2018.2803828, ISSN: 0162-8828, 2018.



Constant-time Calculation of Zernike Moments for Detection with Rotational Invariance

Aneta Bera, Przemysław Klęsk and Dariusz Sychel

Abstract—We construct a set of special complex-valued integral images and an algorithm that allows to calculate Zernike moments fast, namely in constant time. The technique is suitable for dense detection procedures, where the image is scanned by a sliding window at multiple scales, and where rotational invariance is required at the level of each window. We assume no preliminary image segmentation. Owing to the proposed integral images and binomial expansions, the extraction of each feature does not depend on the number of pixels in the window and thereby is an $O(1)$ calculation. We analyze algorithmic properties of the proposition, such as: number of needed integral images, complex-conjugacy of integral images, number of operations involved in feature extraction, speed-up possibilities based on lookup tables. We also point out connections between Zernike and orthogonal Fourier–Mellin moments in the context of computations backed with integral images. Finally, we demonstrate three examples of detection tasks of varying difficulty. Detectors are trained on the proposed features by the RealBoost algorithm. When learning, the classifiers get acquainted only with examples of target objects in their upright position or rotated within a limited range. At the testing stage, generalization onto the full 360° angle takes place automatically.

Index Terms—complex-valued integral images, Zernike moments, detection, rotational invariance, constant-time feature extraction

...

- **Impact Factor: 8.3, #1-ranked journal**

IEEE TRANSACTIONS ON
PATTERN ANALYSIS AND
MACHINE INTELLIGENCE

Computer Vision and Pattern Recognition (1/66),
Computational Theory and Mathematics (1/97),
Software (1/367),
Artificial Intelligence (1/152),
Applied Mathematics (1/398)

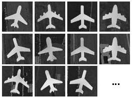
High-level intuition — scenario A

Scenario A: standalone detector invariant to rotation

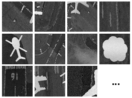
TRAIN

positives upright

±45° random rotations:



negatives:



↓

machine learning →

TEST

generalization onto any rotation angle (360°)

test input image

↓

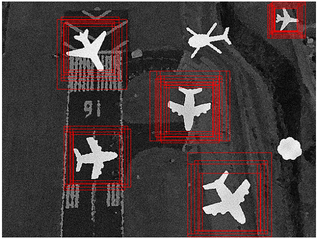
complex-valued integral images

↓

detection procedure

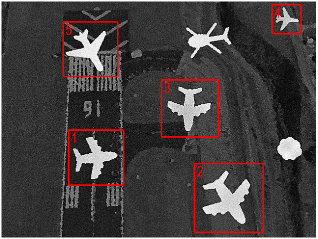
(constant-time features invariant to rotation based on ZMs)

↳



detector (red)

→



after postprocessing (grouping)

High-level intuition — scenario B

Scenario B: prescreener invariant to rotation + angle-dependent classifiers

TRAIN

positives upright

±45° random rotations:

...

negatives:

...

↓

machine learning →

TEST
generalization onto any rotation angle (360°)

test input image

↓

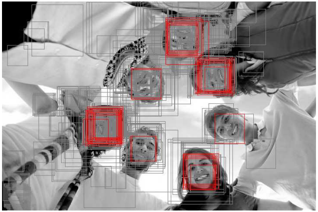
complex-valued integral images

↓

detection procedure

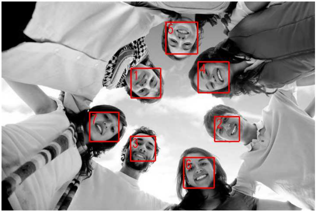
(constant-time features invariant to rotation based on ZMs)

↳



prescreener (gray)+angle-dependent classifs. (gray→red)

→



after postprocessing (grouping)

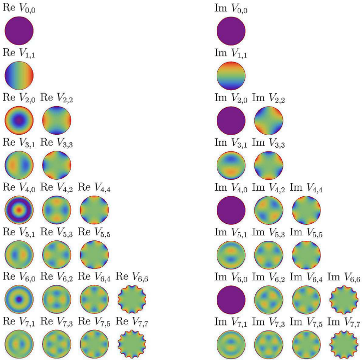
Zernike polynomials and moments

- **Zernike polynomials (ZPs):** set of orthogonal, complex-valued functions over unit disk in polar coordinates (F. Zernike, 1934).
- Products of: standard polynomials over radius and harmonic terms over angle.
- Obtained via G-S orthogonalization for: $\{1, re^{i\theta}, r^2, r^2e^{2i\theta}, r^3e^{i\theta}, r^3e^{3i\theta}, r^4, r^4e^{2i\theta}, r^4e^{4i\theta}, \ldots\}$.

$$
\begin{aligned}
V_{p,q}(r,\theta) &= R_{p,q}(r)F_q(\theta) \\
&= \sum_{s=0}^{(p-|q|)/2} \beta_{p,q,s} r^{p-2s} e^{iq\theta},
\end{aligned}
$$

where

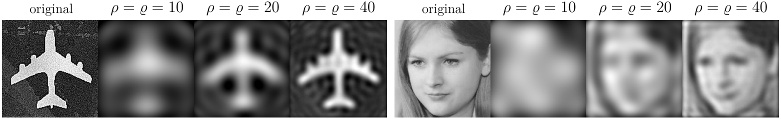
$$
\beta_{p,q,s} = \frac{(-1)^s (p-s)!}{s!\,((p+q)/2-s)!\,((p-q)/2-s)!}.
$$



Zernike polynomials and moments

- **Zernike moments (ZMs):** coefficients of expansion (of function / image) in terms of ZPs.

$$i(r,\theta) = \sum_{0\leqslant p\leqslant\infty} \sum_{\substack{-p\leqslant q\leqslant p\\ p-|q|\ \text{even}}} M_{p,q}V_{p,q}(r,\theta) \approx \sum_{0\leqslant p\leqslant\rho} \sum_{\substack{-\min\{p,\varrho\}\leqslant q\leqslant\min\{p,\varrho\}\\ p-|q|\ \text{even}}} M_{p,q}V_{p,q}(r,\theta) \qquad (85)$$



- Optimal coefficients — **ZMs**:

$$M_{p,q} = \frac{p+1}{\pi}\int_0^{2\pi}\int_0^1 i(r,\theta)\sum_{\substack{-p\leqslant q\leqslant p\\ p-|q|\ \text{even}}} \beta_{p,q,s}r^{p-2s}e^{-iq\theta}r\,dr\,d\theta. \qquad (86)$$

(moduli invariant to rotation)

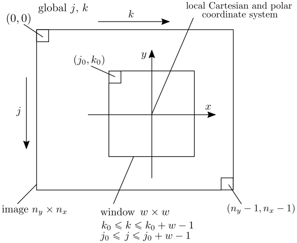
Zernike moments for image windows

- Discrete version of **ZMs for image window** of size $w \times w$:

$$M_{2p+o,2q+o} \approx \widehat{M}_{2p+o,2q+o} = \frac{2(p+1)}{\pi w^2} \sum_{\substack{0\leqslant j\leqslant w-1\\ 0\leqslant k\leqslant w-1}} \sum i(j,k) \sum_{2q+o\leqslant 2s+o\leqslant 2p+o} \beta_{2p+o,2q+o,p-s}(x_k+iy_j)^{s-q}(x_k-iy_j)^{s+q+o}, \tag{87}$$

where: $x_k = \frac{2k-(w-1)}{w\sqrt{2}}$, $y_j = \frac{w-1-2j}{w\sqrt{2}}$, $0 \leqslant k, j \leqslant w-1$.

- ***Can ZMs be backed with integral images and applied in a detection procedure?***


(0, 0)
global j, k
k
local Cartesian and polar coordinate system
(j_0, k_0)
y
x
j
image $n_y \times n_x$
window $w \times w$
$k_0 \leqslant k \leqslant k_0 + w - 1$
$j_0 \leqslant j \leqslant j_0 + w - 1$
$(n_y - 1, n_x - 1)$

- ***Is it possible to design such cumulants in the single global system of Cartesian coordinates that shall later allow for extracting features (invariant to rotation) in many local systems of polar coordinates at the level of each window?***

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

ZMs — our main result

Proposition 2

Suppose a set of integral images $\{ii_{t,u}\}$, *defined as in* (75), *has been prepared prior to the detection procedure. Then, for any square window in the image, each of its Zernike moments can be calculated in constant time —* $O(1)$, *regardless of the number of pixels in the window, as follows:*

$$\widehat{M}_{2p+o,2q+o} = \frac{4p+2o+2}{\pi w^2} \sum_{2q+o\leqslant 2s+o\leqslant 2p+o} \beta_{2p+o,2q+o,p-s}\left(\frac{\sqrt{2}}{w}\right)^{2s+o}$$

$$\cdot \sum_{t=0}^{s-q}\binom{s-q}{t}(-k_c+ij_c)^{s-q-t}\sum_{u=0}^{s+q+o}\binom{s+q+o}{u}(-k_c-ij_c)^{s+q+o-u}\,\underset{\substack{j_0,j_0+w-1\\k_0,k_0+w-1}}{\Delta}\left(ii_{t,u}\right), \qquad (88)$$

where w *denotes the width and* (j_c, k_c) *the central index of the window.*

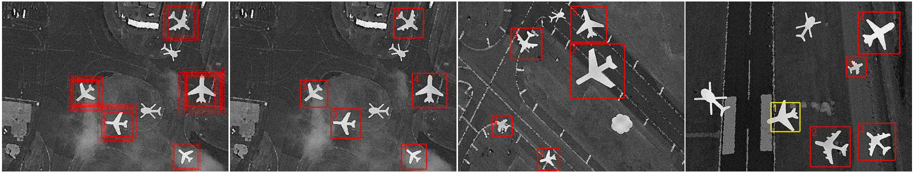
Experimental setup

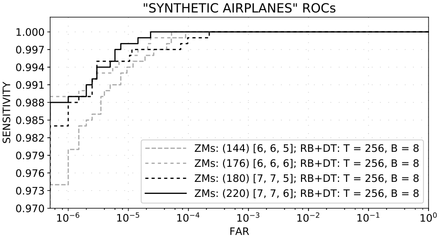
Typical settings:

- **Learning algorithm: RealBoost + bins** or **RealBoost + decision trees**.
- **Detectors: ensembles of 256** or **512 classifiers**.
- Training set sizes: ≈ 10 000 positives, 100 000 negatives.
- Training duration: ≈ 2 h to 6 h.
- Image / video resolution: 640 × 480.
- **Detection procedure (heavy):** ≈ 151 000 windows
  (8 scales, sliding window 48 × 48 up to 172 × 172, jumps ratio 0.05).
- **Detection procedure (light):** ≈ 11 000 windows
  (4 scales, sliding window 120 × 120 up to 207 × 207, jumps ratio 0.05).
- **Software: C#** and **C++** (crucial computational procedures as dlls).

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

“Synthetic airplanes”

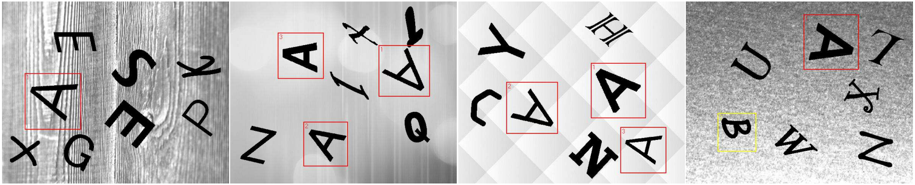


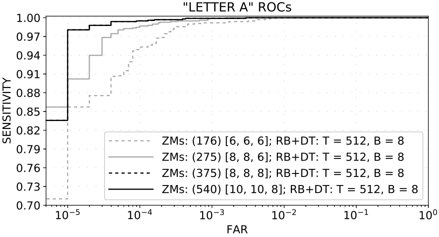


ZMs (220) [7, 7, 6], $B = 8$, $T = 256$	
quantity (or operations)	time or amount
no. of analyzed windows	150 849
total time of detection procedure	847 ms
no. of prepared integral images	20
preparation time of all integral images (complex-valued)	106 ms
preparation time per 1 integral image	5.3 ms
time per 1 window	5.615 μs (amortized: 4.912 μs)
no. of distinct features used by ensemble	219
time per 1 window and 1 feature	25.64 ns (amortized: 22.43 ns)

“Synthetic airplanes”: time performance for a 640 × 480 image
(parallel computations on: Intel Xeon E5-2699 v4 CPU, 22/44 c/t, 55 MB cache).

“Letter A”



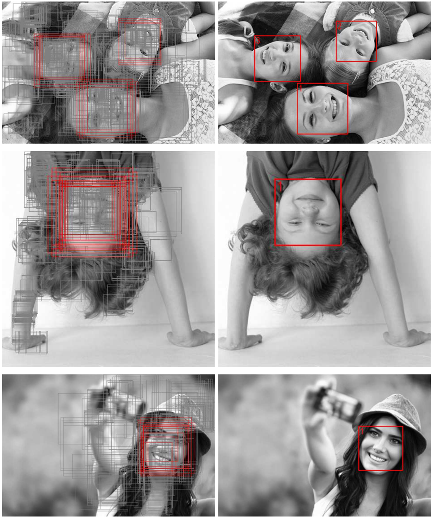

"LETTER A" ROCs
SENSITIVITY
1.00
0.97
0.94
0.91
0.88
0.85
0.82
0.79
0.76
0.73
0.70
FAR
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
ZMs: (176) [6, 6, 6]; RB+DT: T = 512, B = 8
ZMs: (275) [8, 8, 6]; RB+DT: T = 512, B = 8
ZMs: (375) [8, 8, 8]; RB+DT: T = 512, B = 8
ZMs: (540) [10, 10, 8]; RB+DT: T = 512, B = 8

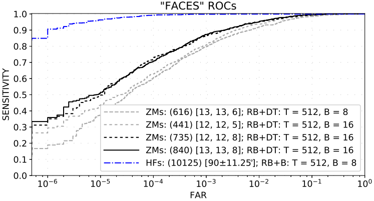
ZMs (540) [10, 10, 8], $B = 8$, $T = 512$	
quantity (or operations)	time or amount
no. of analyzed windows	18 588
total time of detection procedure	289 ms
no. of prepared integral images	25
preparation time of all integral images (complex-valued)	132.8 ms
preparation time per 1 integral image	5.31 ms
time per 1 window	15.5 μs (amortized: 8.42 μs)
no. of distinct features used by ensemble	375
time per 1 window and 1 feature	41.46 ns (amortized: 22.45 ns)

“Letter A”: time performance for a 640 × 480 image
(parallel computations on: Intel Xeon E5-2699 v4 CPU, 22/44 c/t, 55 MB cache).

Research project no.: 2016/21/B/ST6/01495 (National Science Centre, Poland)

"Faces"



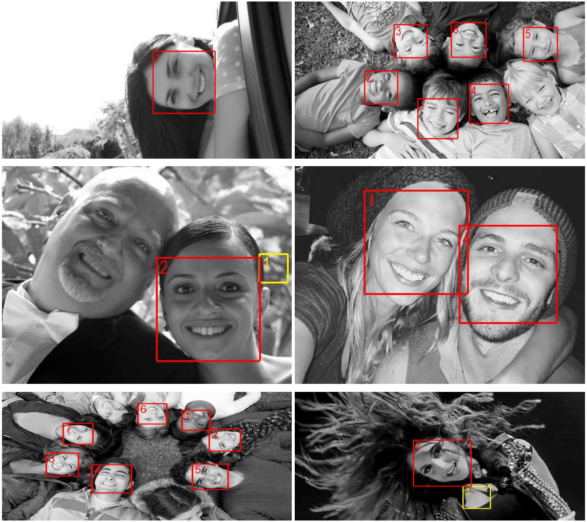

"FACES" ROCs
SENSITIVITY
FAR
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
ZMs: (616) [13, 13, 6]; RB+DT: T = 512, B = 8
ZMs: (441) [12, 12, 5]; RB+DT: T = 512, B = 16
ZMs: (735) [12, 12, 8]; RB+DT: T = 512, B = 16
ZMs: (840) [13, 13, 8]; RB+DT: T = 512, B = 16
HFs: (10125) [90±11.25°]; RB+B: T = 512, B = 8

ZMs (840) [13, 13, 8], $B = 16$, $T = 512$	
quantity (or operations)	time or amount
no. of analyzed windows	150 849
total time of detection procedure	2 543 ms
no. of prepared integral images	56
preparation time of all integral images (complex-valued)	263 ms
preparation time per 1 integral image	4.7 ms
time per 1 window	16.86 μs (amortized: 15.11 μs)
no. of distinct features used by ensemble	331
time per 1 window and 1 feature	50.93 ns (amortized: 45.66 ns)
total time of 16 angular classifiers	294 ms

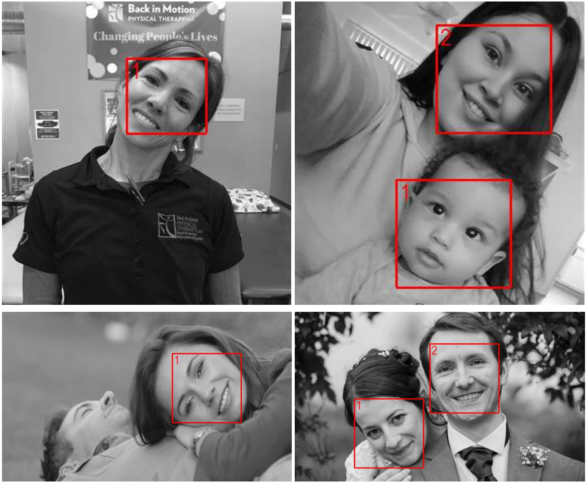
vs

16 complete scans with HFs (1 025), $B = 8$, $T = 512$	
no. of analyzed windows in total	2 413 584
total time 16 detection procedures	4 752 ms
preparation time of 1 integral image	6 ms
time per 1 detection procedure	297 ms
time per 1 window	1.97 μs
average no. of distinct features used by ensemble	495
time per 1 window and 1 feature	3.98 ns

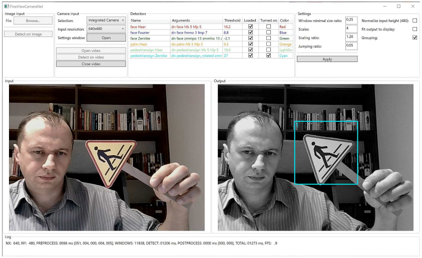
“Faces”



“Faces”


Back in Motion
PHYSICAL THERAPY LLC
Changing People's Lives
1
2
1
1
2
1

“Road sign”



File:

Selection: Integrated Camera

Input resolution: 640x480

Settings window: Open

Name	Arguments	Threshold	Loaded	Turned on	Color
face Haar	dn face hfs 5 hfp 5	16.2	☑	☐	Red
face Fourier	dn face fmmo 3 fmp 7	8.8	☑	☐	Blue
face Zernike	dn face zmmpo 13 zmmho 13 z	-2.1	☑	☐	Green
palm Haar	dn palm hfs 5 hfp 5	9.3	☑	☐	Orange
pedestriansign Haar	dn pedestriansign hfs 5 hfp 5	19.4	☑	☐	LightGr
pedestriansign Zernike	dn pedestriansign_rotated zmn	27	☑	☑	Cyan

Window minimal size ratio: 0.25

Scales: 4

Scaling ratio: 1.20

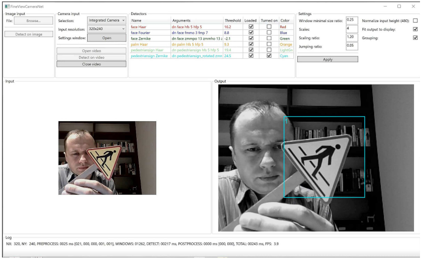
Jumping ratio: 0.05

Normalize input height (480): ☐

Fit output to display: ☐

Grouping: ☑

```
NX:  640, NY:  480, PREPROCESS: 0066 ms [051, 004, 000, 004, 005], WINDOWS: 11838, DETECT: 01206 ms, POSTPROCESS: 0000 ms [000, 000], TOTAL: 01273 ms, FPS:  .9
```



File:

Selection: Integrated Camera

Input resolution: 320x240

Settings window: Open

Name	Arguments	Threshold	Loaded	Turned on	Color
face Haar	dn face hfs 5 hfp 5	16.2	☑	☐	Red
face Fourier	dn face fmmo 3 fmp 7	8.8	☑	☐	Blue
face Zernike	dn face zmmpo 13 zmmho 13 z	-2.1	☑	☐	Green
palm Haar	dn palm hfs 5 hfp 5	9.3	☑	☐	Orange
pedestriansign Haar	dn pedestriansign hfs 5 hfp 5	19.4	☑	☐	LightGr
pedestriansign Zernike	dn pedestriansign_rotated zmn	24.5	☑	☑	Cyan

Window minimal size ratio: 0.25

Scales: 4

Scaling ratio: 1.20

Jumping ratio: 0.05

Normalize input height (480): ☐

Fit output to display: ☑

Grouping: ☑

```
NX:  320, NY:  240, PREPROCESS: 0025 ms [021, 000, 000, 001, 001], WINDOWS: 01262, DETECT: 00217 ms, POSTPROCESS: 0000 ms [000, 000], TOTAL: 00243 ms, FPS:  3.9
```

ZMs — conclusions

1. **Algorithmic result for detection tasks:** a computational technique for **constant-time extraction of Zernike moments**, backed with **complex-valued integral images**.
2. Suitable for **detection procedures** where **rotational invariance** is a requirement.
3. Proposed additional refinements: **complex-conjugacy of integral images**, **speed-up possibilities using LUTs**.
4. Equivalent representation for **Fourier–Mellin moments *not* feasible**.
5. Future work: analysis of numerical errors, product invariants of form: $M_{p,q}^{k} \cdot M_{r,s}$, $kq + s = 0$.

References

Bera, A., Klęsk, P., & Sychel, D. (2018). Constant-time Calculation of Zernike Moments for Detection with Rotational Invariance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. doi: 10.1109/TPAMI.2018.2803828

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Monterey, CA, USA: Wadsworth & Brooks.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Conference on Computer Vision and Pattern Recognition (CVPR'2005)* (Vol. 1, pp. 886–893). San Diego, CA, USA: IEEE.

Freund, Y. (1995). Boosting a Weak Learning Algorithm by Majority. *Informationa and Computation*, *121*(2), 256–285.

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. In *Machine learning: Proceedings of the thirteenth international conference* (pp. 148–156). Morgan Kaufman.

Freund, Y., & Schapire, R. (1997). A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting. *Journal of Computer Science and System Sciences*, *55*, 119–139.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, *28*(2), 337–407.

Klęsk, P. (2017). Constant-Time Fourier Moments for Face Detection — Can Accuracy of Haar-Like Features Be Beaten? In *Artificial intelligence and soft computing: 16th international conference, icaisc 2017* (Vol. 10245, pp. 530–543). Springer Int. Publishing.

Klęsk, P., Godziuk, A., Kapruziak, M., & Olech, B. (2015). Fast Analysis of C-scans From Ground Penetrating Radar via 3D Haar-like Features With Application to Landmine Detection. *IEEE Transactions on Geoscience and Remote Sensing*, *53*(7), 3996–4009. doi: 10.1109/TGRS.2015.2388713

Klęsk, P., Godziuk, A., Kapruziak, M., & Olech, B. (2016). Fast Extraction of 3D Fourier Moments via Multiple Integral Images: An Application to Antitank Mine Detection in GPR C-Scans. In L. Chmielewski et al. (Eds.), *Computer Vision and Graphics* (pp. 206–220). Cham: Springer International Publishing.

Klęsk, P., Godziuk, A., Kapruziak, M., & Olech, B. (2017). Statistical moments calculated via integral images in application to landmine detection from Ground Penetrating Radar 3D scans. *Pattern Analysis and Applications*. doi: 10.1007/s10044-016-0592-5

Klęsk, P., & Bera, A. (2018). Constant-time Extraction of Statistical Moments for Object Detection Procedures. In *Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods - volume 1: ICPRAM* (pp. 49–59). SciTePress. doi: 10.5220/0006550000490059

Rasolzadeh, B., et al. (2006). Response Binning: Improved Weak Classifiers for Boosting. In *Ieee intelligent vehicles symposium* (pp. 344–349)