FPGA-GPU Architecture for Kernel SVM Pedestrian Detection

ECVW 2010, San Francisco, 06/13/2010

S. Bauer[1], S. Köhler[2], K. Doll[3], U. Brunsmann[2]

[1] Pattern Recognition Lab (CS 5)
University Erlangen-Nuremberg, Germany

[2] Laboratory for Pattern Recognition and Computational Intelligence
University of Applied Sciences Aschaffenburg, Germany

[3] Laboratory for Computer-Aided Circuit Design
University of Applied Sciences Aschaffenburg, Germany

hochschule aschaffenburg
university of applied sciences

Outline

- Motivation
- HOG/SVM pedestrian detection
- Implementation details
- Evaluation
- Outlook

Motivation | Method | Implementation | Evaluation | Outlook

- Problem scope
  - 1.2 million road traffic deaths per year[1]
  - Majority among vulnerable road users

[1] World Health Organization (WHO), World Bank: World report on road traffic injury. Geneva, 2004.

Motivation | Method | Implementation | Evaluation | Outlook

- **Problem scope**
  - 1.2 million road traffic deaths per year[1]
  - Majority among vulnerable road users

- **State-of-the-art**
  - Optimal performance reported[2,3] on combination of:
    - HOG-like features (Histograms of oriented gradients)
    - Kernel SVM classifier
  - Challenge: computational costs

  - **Idea: Real-time HOG/kernel SVM framework on FPGA-GPU architecture**

[1] World Health Organization (WHO), World Bank: World report on road traffic injury. Geneva, 2004.
[2] P. Dollár et al.: Pedestrian detection: a benchmark. CVPR, 2009.
[3] M. Enzweiler, D. M. Gavrila: Monocular pedestrian detection: survey and experiments. TPAMI, 2009.

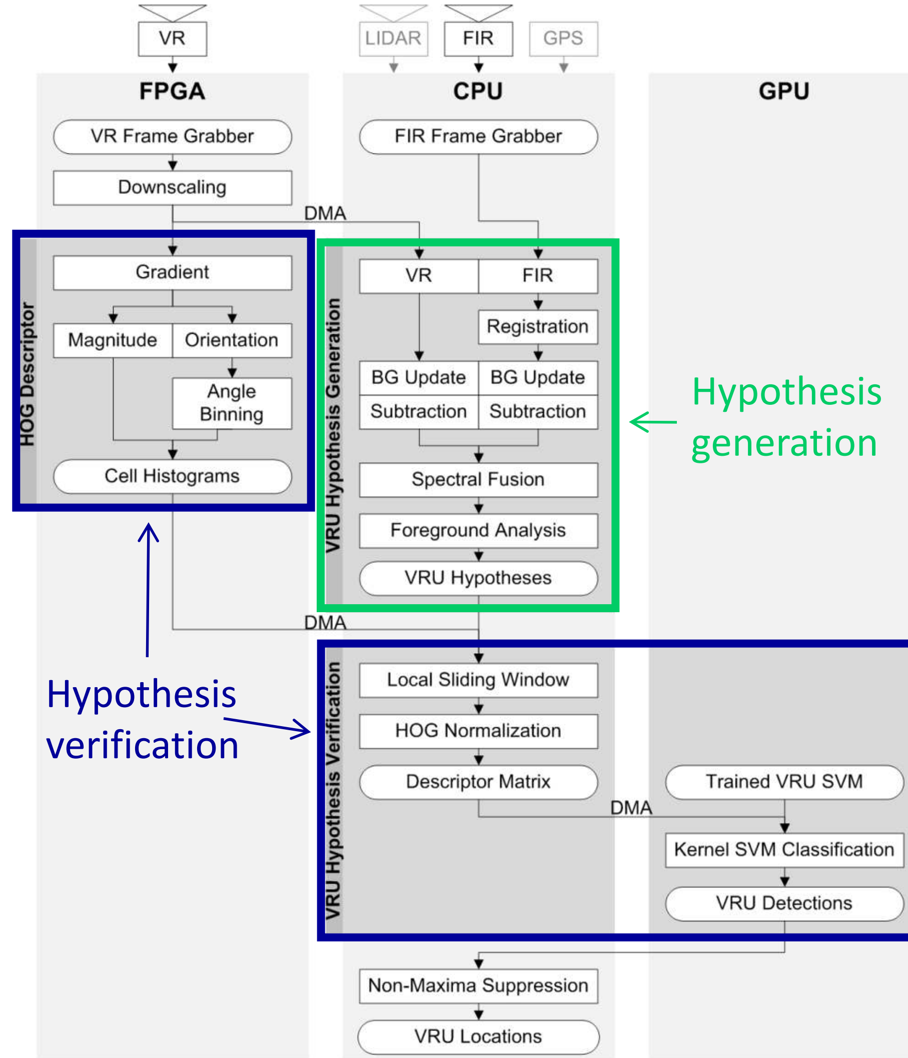
- System overview
  - Road side unit (intersection assistance)

S. Bauer, S. Köhler, K. Doll, U. Brunsmann

System overview

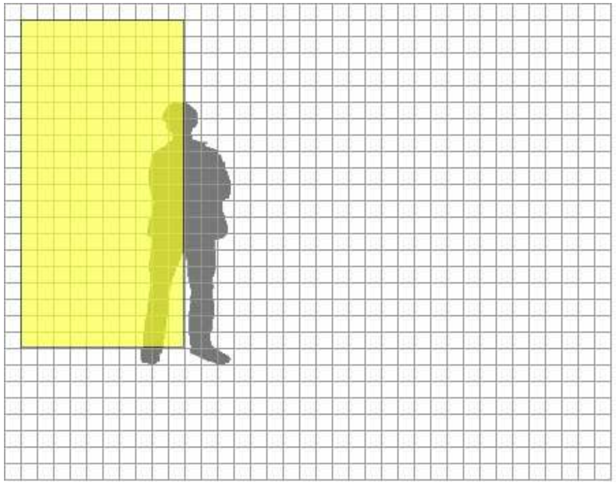
- Road side unit (intersection assistance)

- FPGA-CPU-GPU architecture
  - FPGA: Xilinx Spartan 3 XC3S 4000
  - CPU: Intel Core i7, 2.66 GHz
  - GPU: NVIDIA Gefore GTX 295



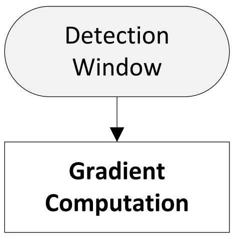
HOG/SVM pedestrian detection:

- Sliding detection window



- For each window:
  - HOG descriptor
  - Kernel SVM classification (pedestrian/non-pedestrian) $f(x) = sgn(\sum_{i=1}^{n_S} \alpha_i y_i K(\boldsymbol{s_i}, \boldsymbol{x}) + b_0)$

HOG descriptor[4]



- Image gradient
  - Point derivatives

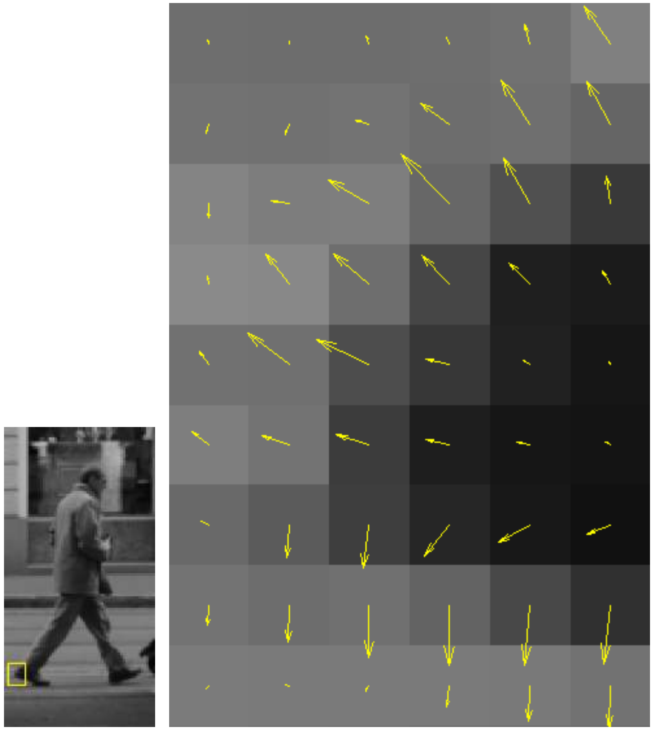
$$G_x = M_x * I \qquad M_x = [-1\ \ 0\ \ 1]$$
$$G_y = M_y * I \qquad M_y = [-1\ \ 0\ \ 1]^T$$

  - Magnitude

$$|G(x,y)| = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$$

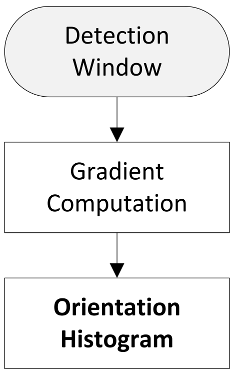
  - Orientation

$$\tan(\phi(x,y)) = \frac{G_y(x,y)}{G_x(x,y)}$$

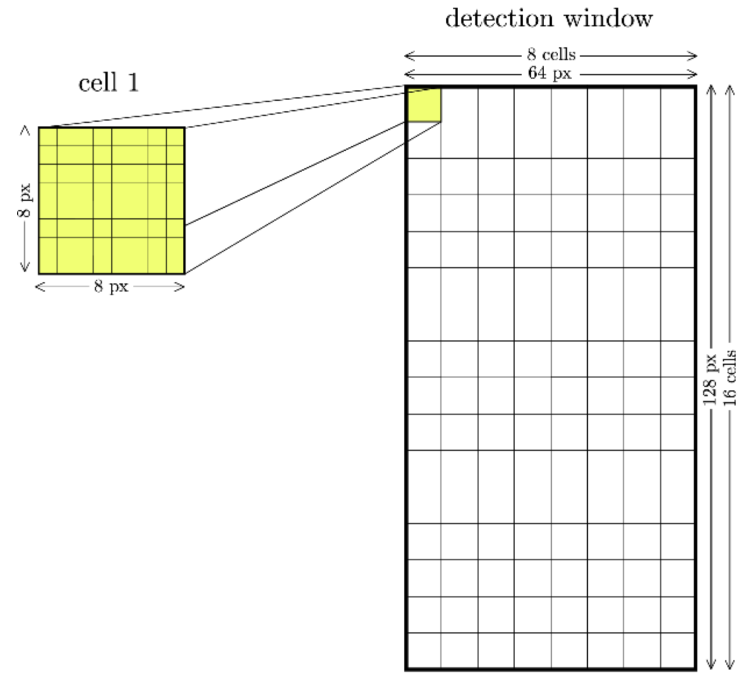


[4] N. Dalal, B. Triggs: Histograms of oriented gradients for human detection. CVPR, 2005.

- HOG descriptor



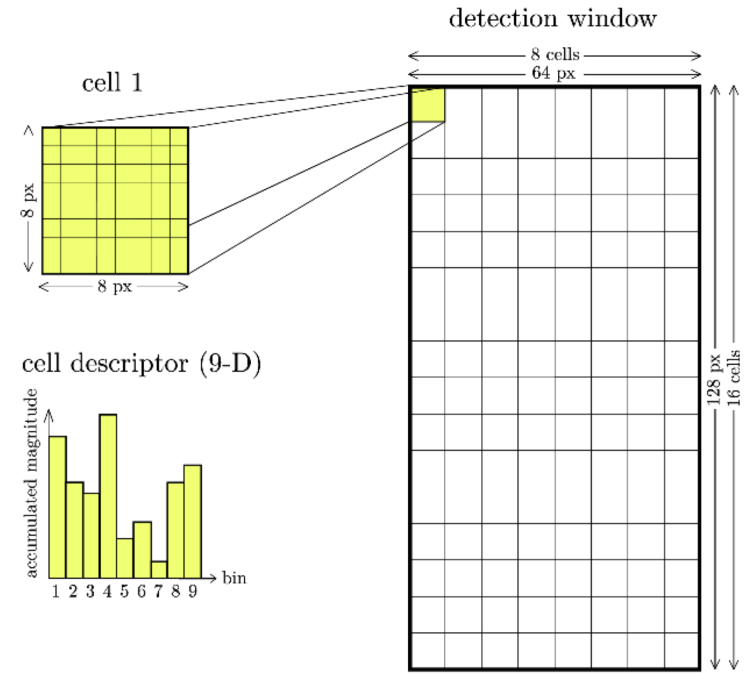
  - Cell descriptor
    - Window subdivision into cells



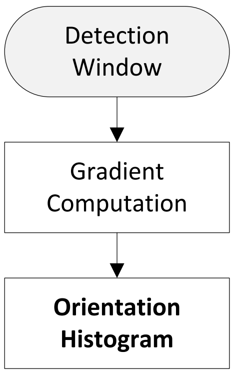
HOG descriptor

Detection Window → Gradient Computation → **Orientation Histogram**

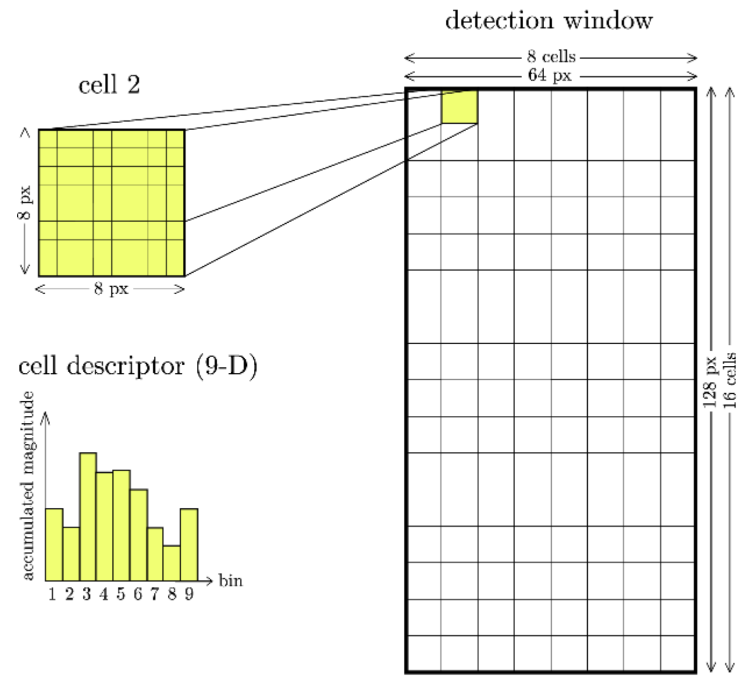
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**



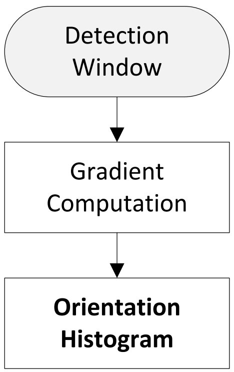
HOG descriptor



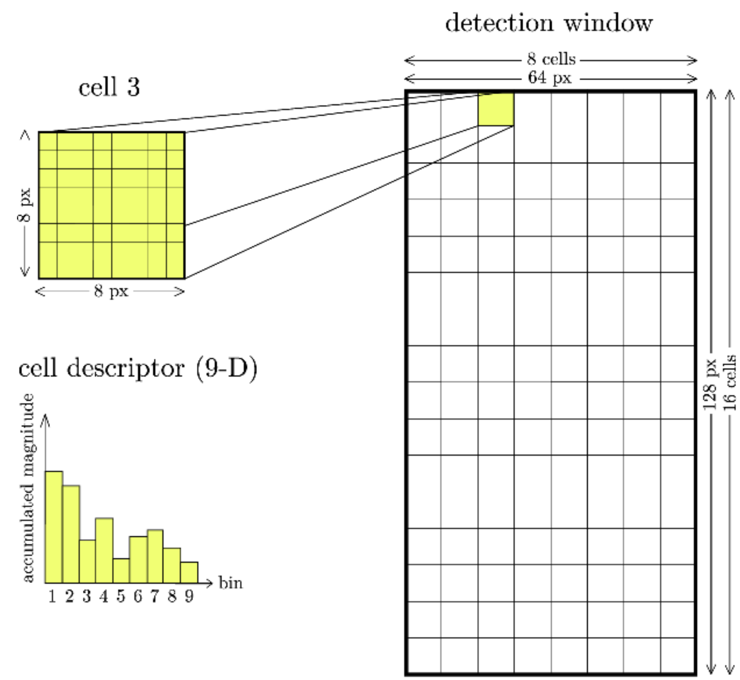
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**



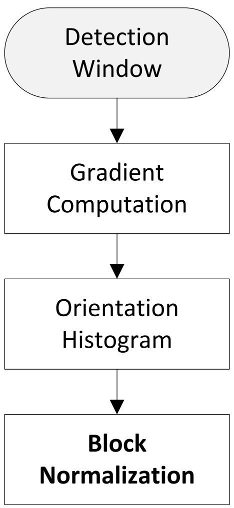
HOG descriptor



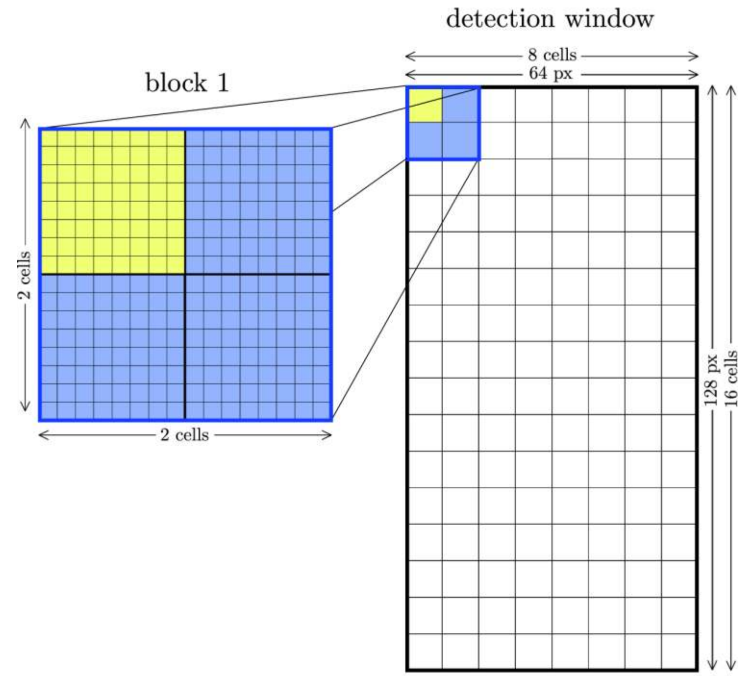
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**



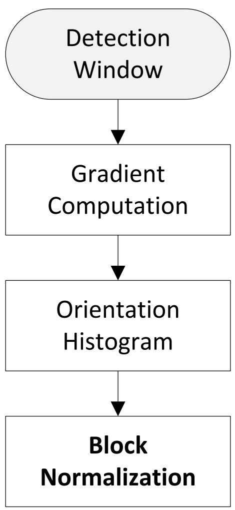
HOG descriptor



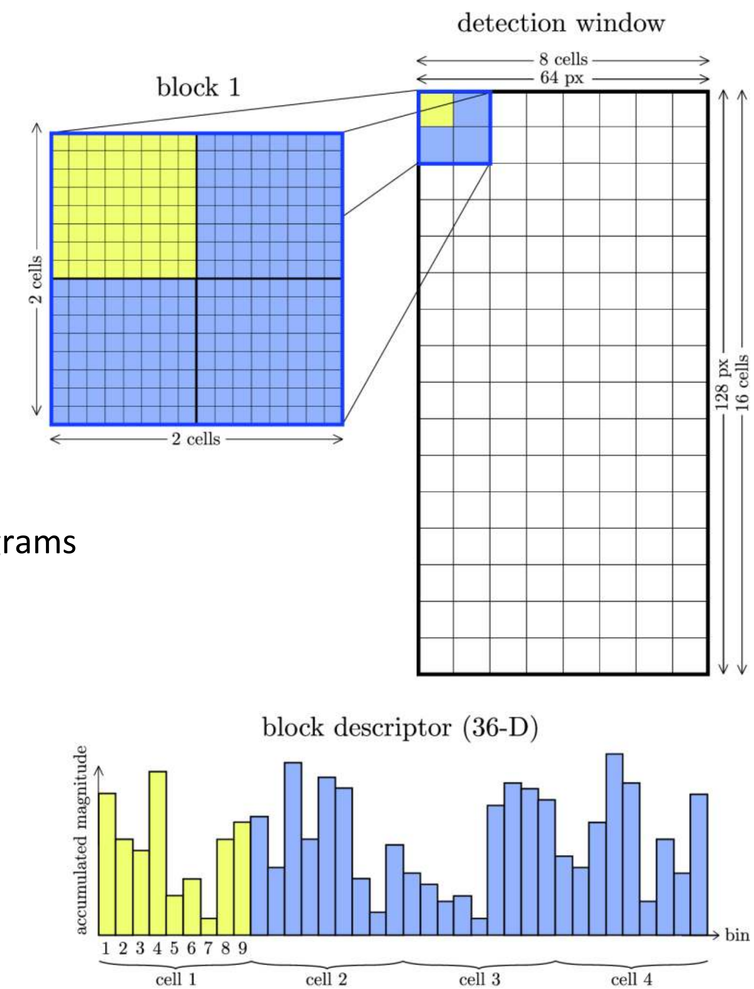
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**
- Block descriptor
  - Subdivision into blocks (2x2 cells)



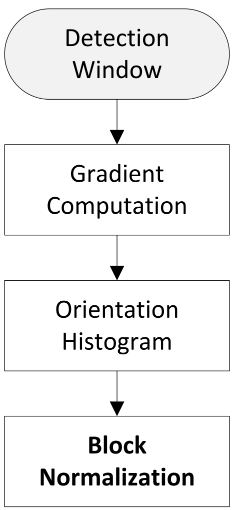
HOG descriptor



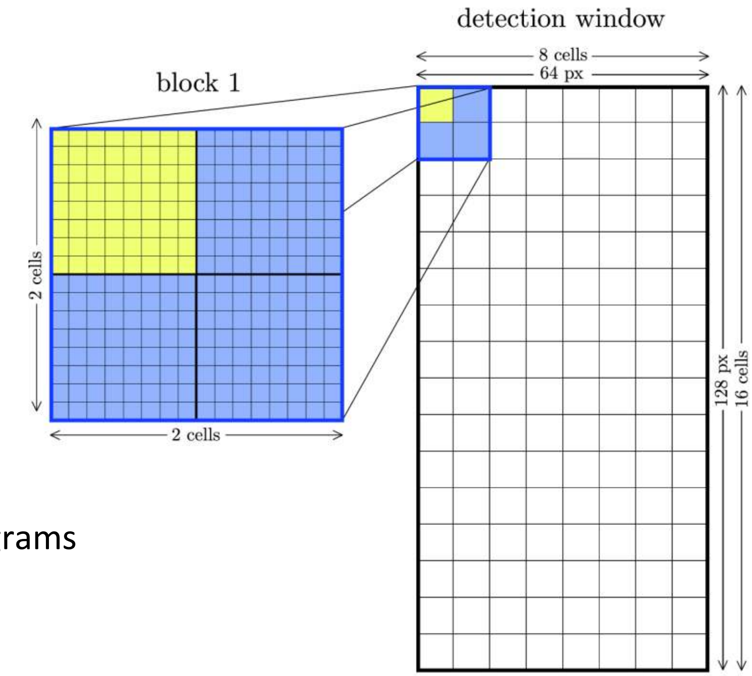
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**
- Block descriptor
  - Subdivision into blocks (2x2 cells)
  - Per block: Concatenation of cell histograms

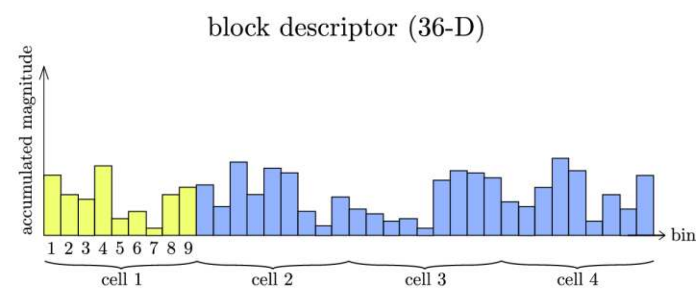


HOG descriptor

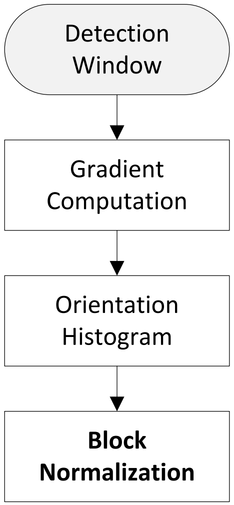


- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**
- Block descriptor
  - Subdivision into blocks (2x2 cells)
  - Per block: Concatenation of cell histograms
  - L2-Normalization: $\boldsymbol{v} \longrightarrow \frac{\boldsymbol{v}}{\sqrt{\|\boldsymbol{v}\|_2^2 + \epsilon^2}}$

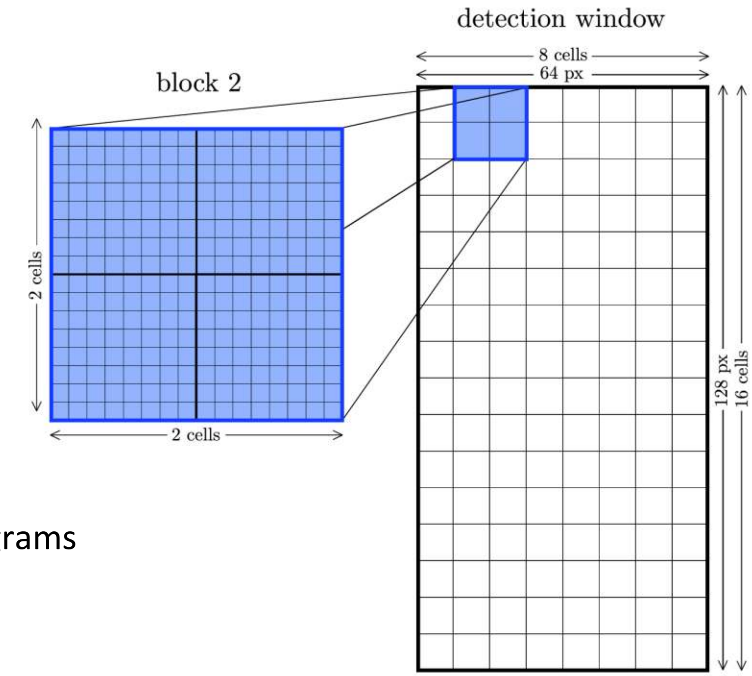


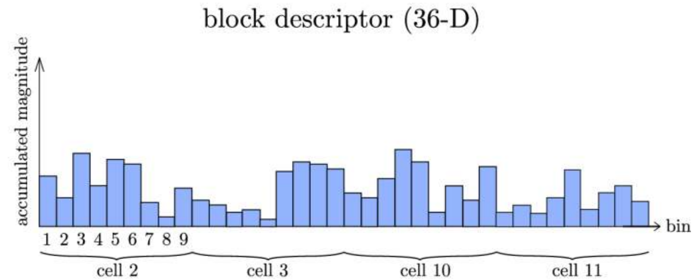


HOG descriptor

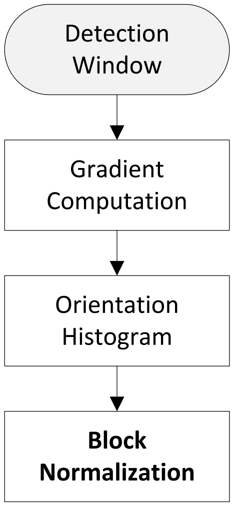


- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**
- Block descriptor
  - Subdivision into blocks (2x2 cells)
  - Per block: Concatenation of cell histograms
  - L2-Normalization: $\boldsymbol{v} \longrightarrow \frac{\boldsymbol{v}}{\sqrt{\|\boldsymbol{v}\|_2^2 + \epsilon^2}}$

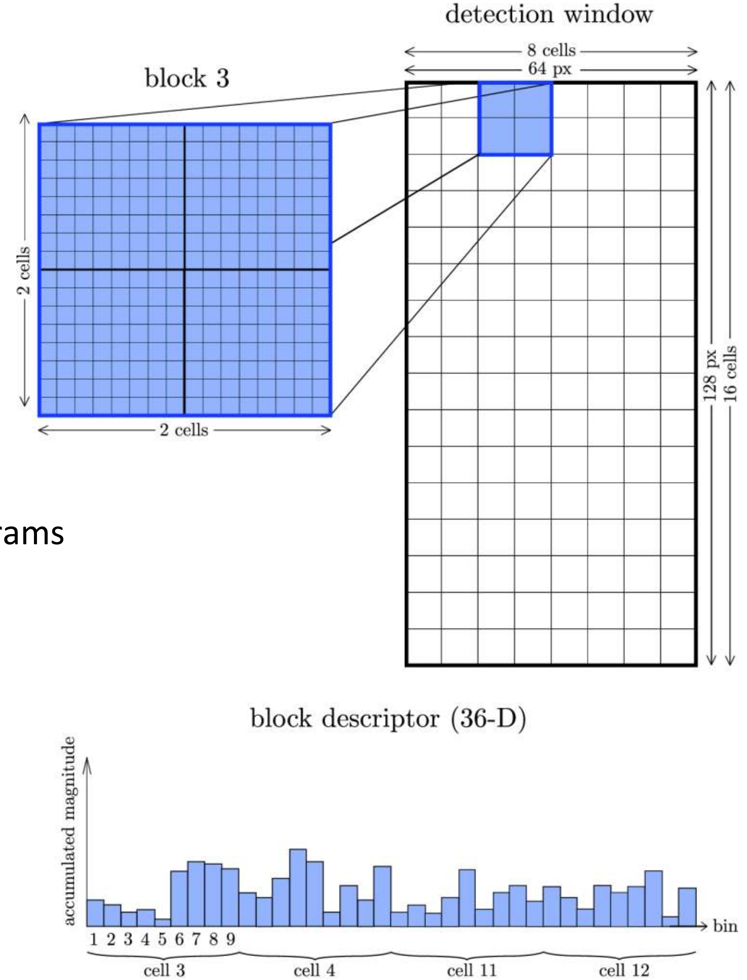




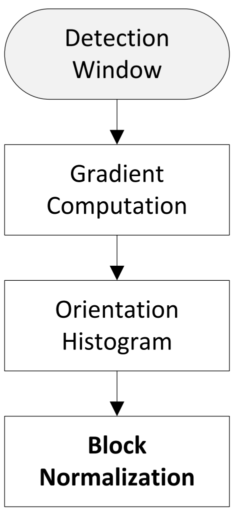
HOG descriptor



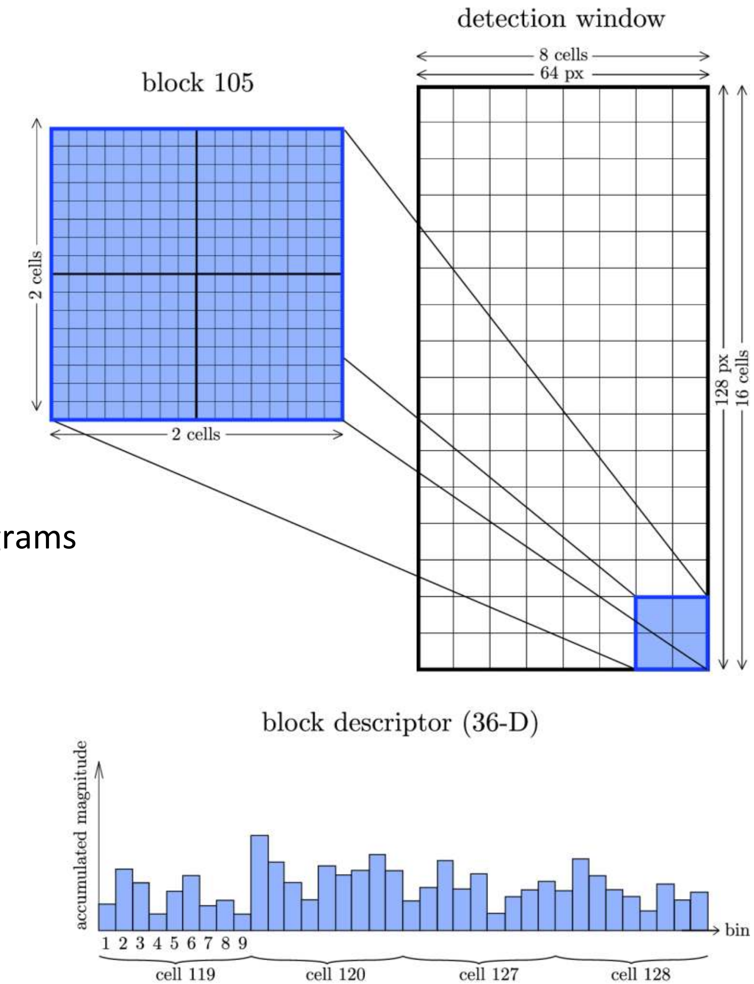
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**
- Block descriptor
  - Subdivision into blocks (2x2 cells)
  - Per block: Concatenation of cell histograms
  - L2-Normalization: $\boldsymbol{v} \longrightarrow \frac{\boldsymbol{v}}{\sqrt{\|\boldsymbol{v}\|_2^2 + \epsilon^2}}$



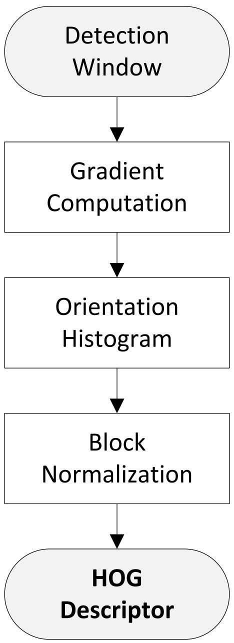
HOG descriptor



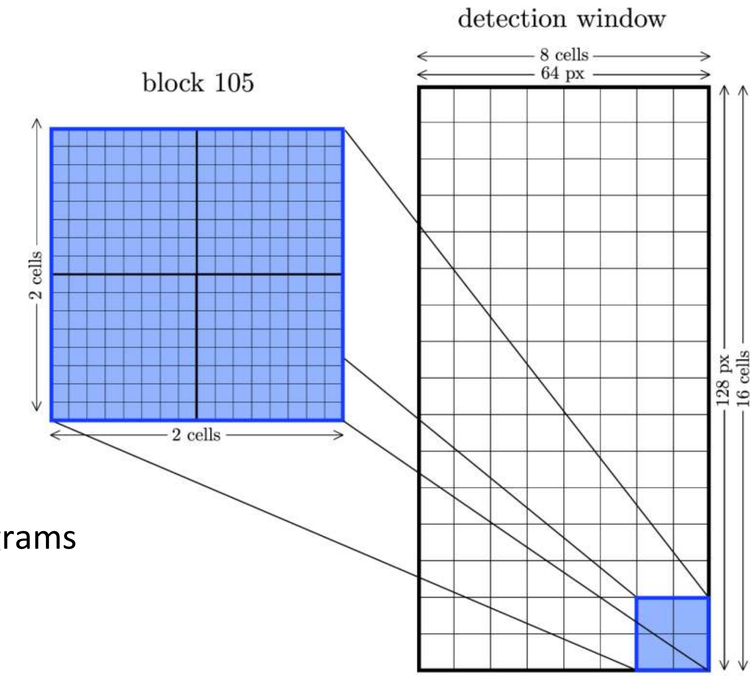
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**
- Block descriptor
  - Subdivision into blocks (2x2 cells)
  - Per block: Concatenation of cell histograms
  - L2-Normalization: $\boldsymbol{v} \longrightarrow \frac{\boldsymbol{v}}{\sqrt{\|\boldsymbol{v}\|_2^2 + \epsilon^2}}$

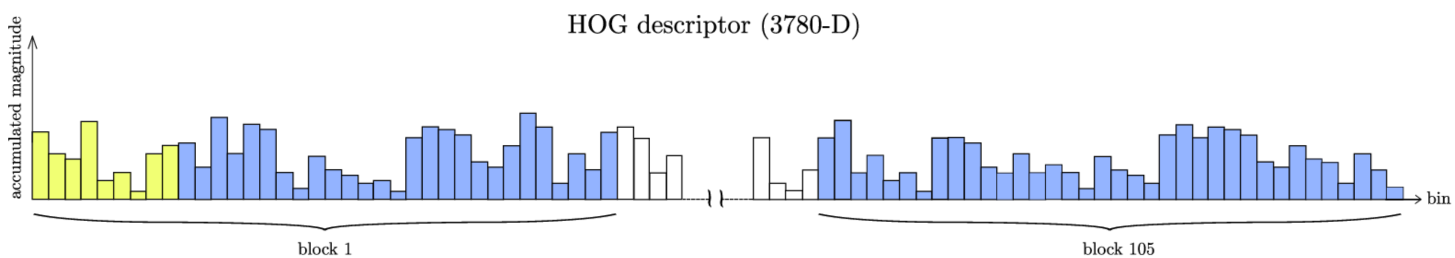


HOG descriptor

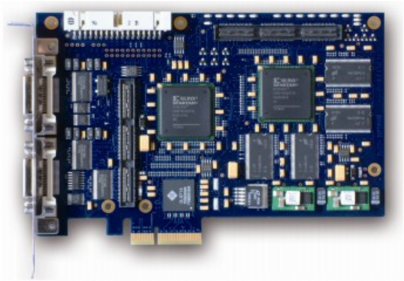


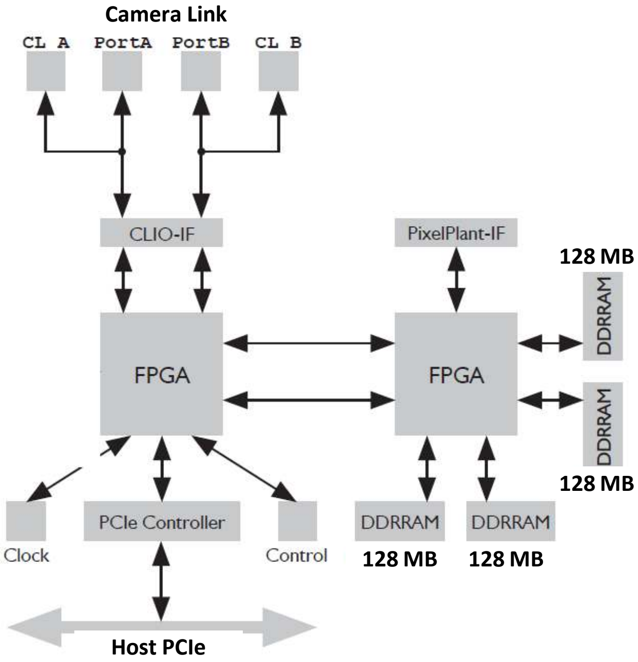
- Cell descriptor
  - Window subdivision into cells
  - Per cell: **Orientation histogram**
- Block descriptor
  - Subdivision into blocks (2x2 cells)
  - Per block: Concatenation of cell histograms
  - L2-Normalization: $\boldsymbol{v} \longrightarrow \frac{\boldsymbol{v}}{\sqrt{\|\boldsymbol{v}\|_2^2 + \epsilon^2}}$
- HOG descriptor





- FPGA-HOG
  - Rapid prototyping platform: microEnable IV-FULLx4[5]
    - Commercially available PCIe frame grabber board
    - CameraLink interface, FPGA for image (pre-)processing



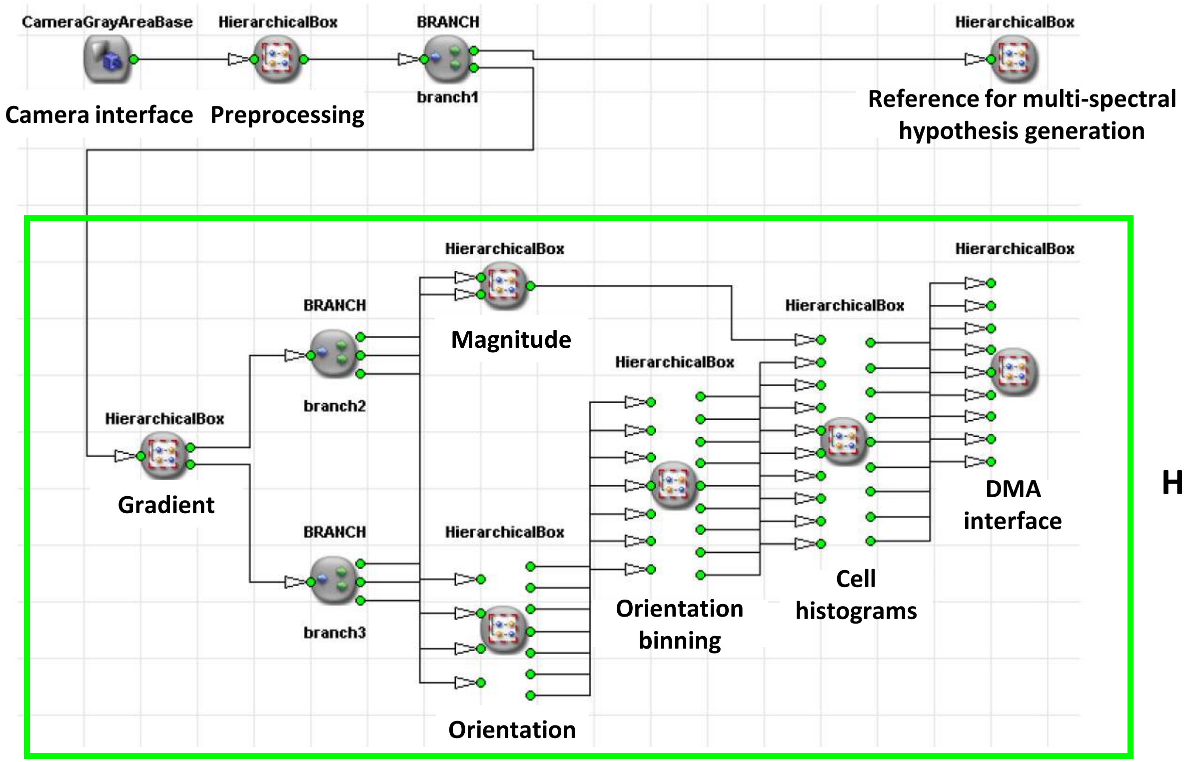


  - VisualApplets (VA)
    - Graphic-oriented hardware development, in combination with Xilinx ISE tools
    - Design arranged by image processing operators and transport links

[5] http://www.silicon-software.com

FPGA-HOG

- VA flowchart

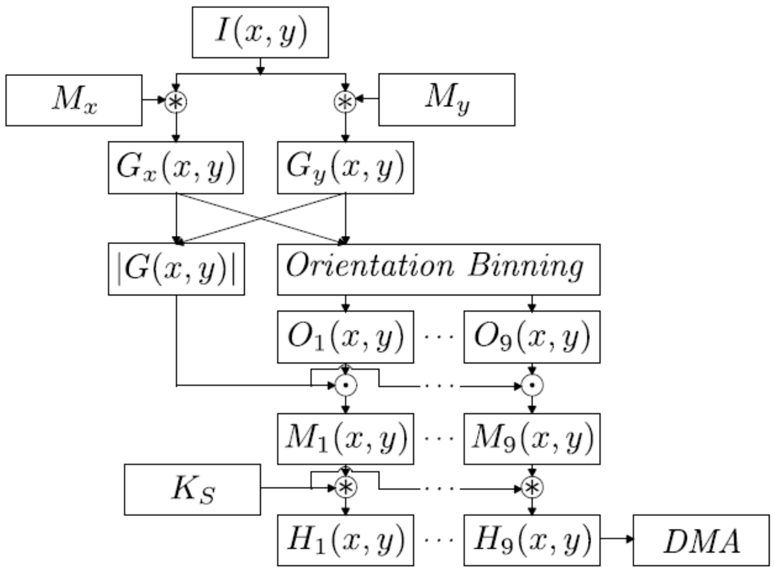


HOG cell descriptors

FPGA-HOG

- Convolution scheme for HOG cell descriptors:

$I(x,y)$	Grayscale input image
M_x, M_y	Point derivative filter kernels
G_x, G_y	Point derivatives
$\lvert G(x,y)\rvert$	Gradient magnitude
$O_i(x,y)$	Orientation image (binary)
$M_i(x,y)$	Magnitude weighted orientation image
K_S	Sum filter kernel
$H_i(x,y)$	Histogram bin image

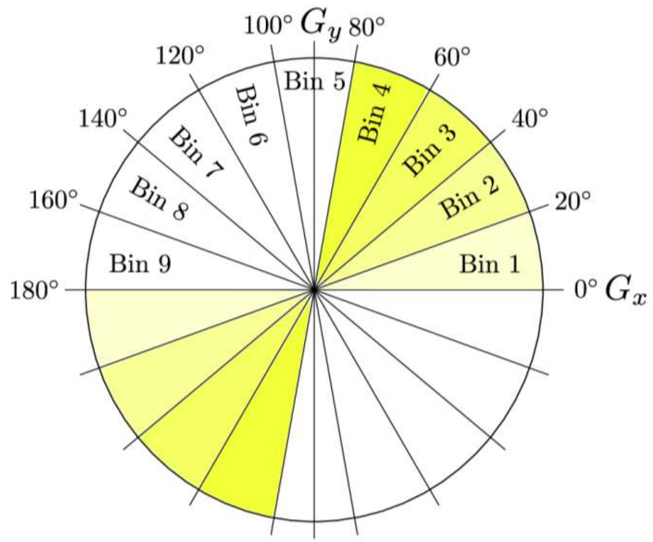


FPGA-HOG

- Gradient magnitude $|G(x,y)| = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$
- Orientation binning
  - Example: Bin 1 (0°-20°)

$$\begin{aligned} Cond.I: \quad & G_x(x,y) > 0 \wedge G_y(x,y) > 0 \Leftrightarrow quadrant\ I \\ \vee \quad & G_x(x,y) < 0 \wedge G_y(x,y) < 0 \Leftrightarrow quadrant\ III \end{aligned}$$

$$\begin{aligned} Cond.II: \quad 0 < & \tan(\alpha) < \tan(20°) \\ 0 < & \left|\frac{G_y(x,y)}{G_x(x,y)}\right| < \tan(20°) \\ 0 < & |G_y(x,y)| < \tan(20°) \cdot |G_x(x,y)| \\ 0 < & |G_y(x,y)| < 0.364 \cdot |G_x(x,y)| \end{aligned}$$



S. Bauer, S. Köhler, K. Doll, U. Brunsmann

- **FPGA-HOG**
  - Orientation histogram
    - Orientation binning
      - O1: 0-60°
      - O2: 60-120°
      - O3: 120-180°

Example:

4x4 px Cell

3 orientation bins

ϕ

47°	21°	78°	170°
98°	94°	5°	140°
27°	77°	30°	128°
4°	11°	22°	101°

O_1

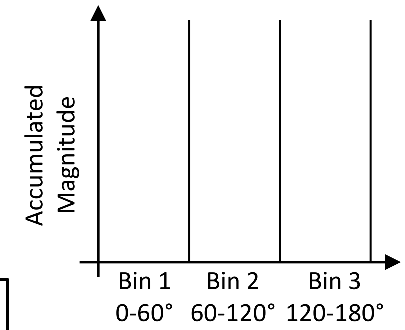
1	1	0	0
0	0	1	0
1	0	1	0
1	1	1	0

O_2

0	0	1	0
1	1	0	0
0	1	0	0
0	0	0	1

O_3

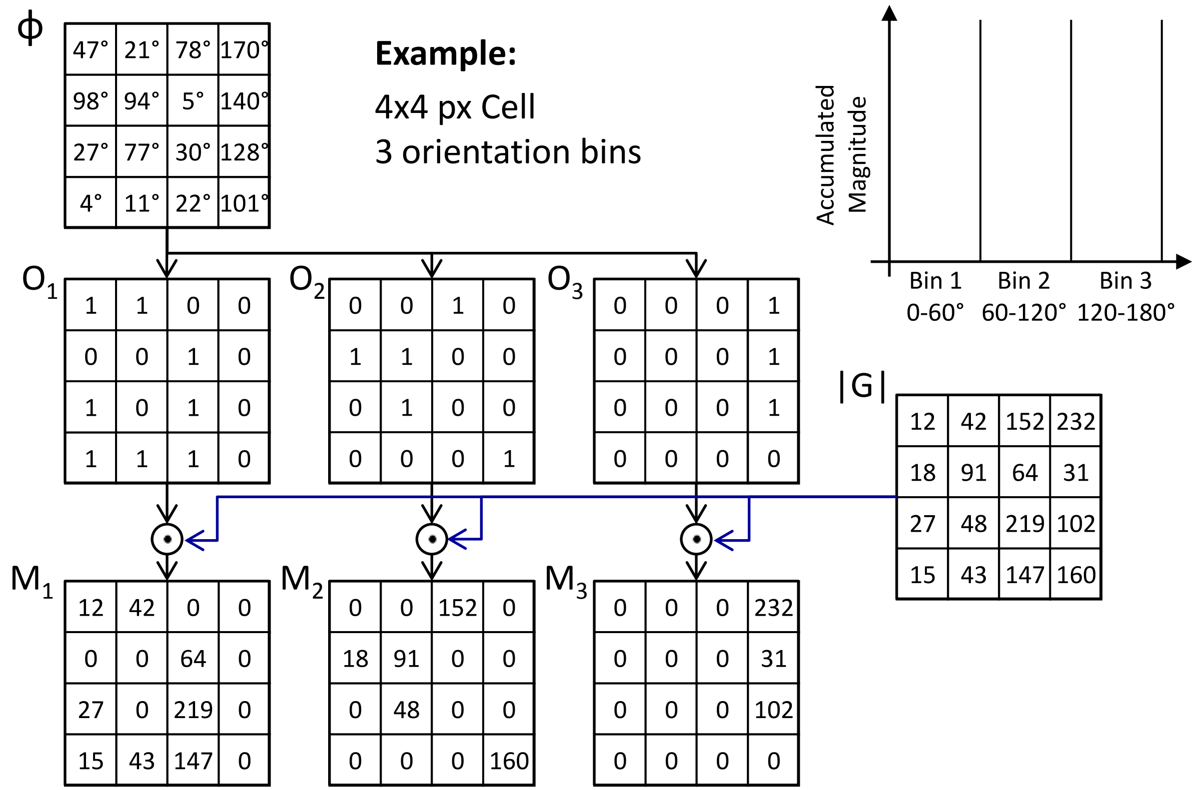
0	0	0	1
0	0	0	1
0	0	0	1
0	0	0	0



|G|

12	42	152	232
18	91	64	31
27	48	219	102
15	43	147	160

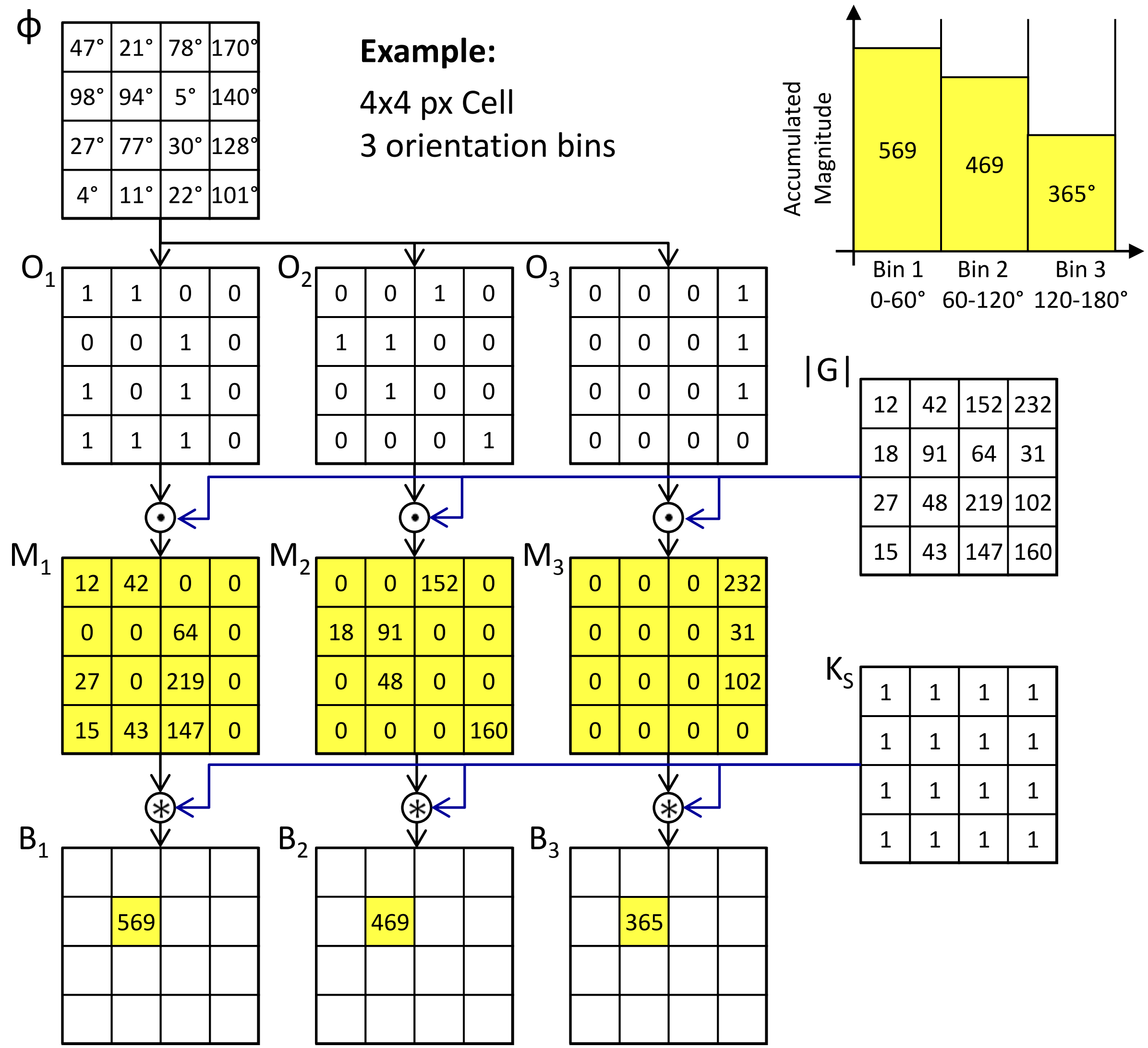
- **FPGA-HOG**
  - Orientation histogram
    - Orientation binning
      - O1: 0-60°
      - O2: 60-120°
      - O3: 120-180°
    - Magnitude weighting



FPGA-HOG

- Orientation histogram
  - Orientation binning
    - O1: 0-60°
    - O2: 60-120°
    - O3: 120-180°
  - Magnitude weighting
  - Bin accumulation

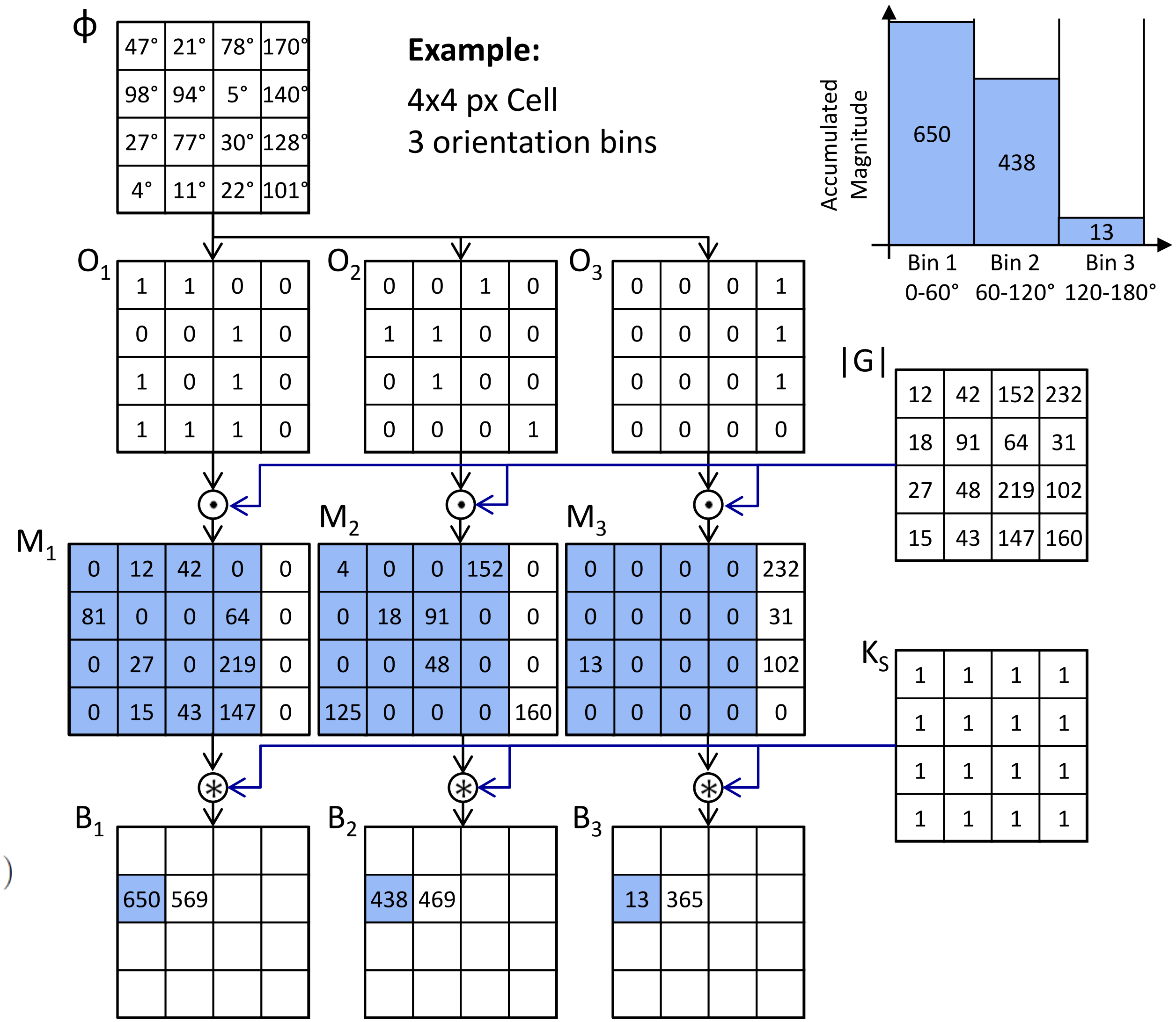
$$H_i = \boldsymbol{k}_S^T * (\boldsymbol{k}_S * M_i)$$
$$\boldsymbol{k}_S = (1 \quad \cdots \quad 1)$$



S. Bauer, S. Köhler, K. Doll, U. Brunsmann

- FPGA-HOG
  - Orientation histogram
    - Orientation binning
      - O1: 0-60°
      - O2: 60-120°
      - O3: 120-180°
    - Magnitude weighting
    - Bin accumulation

$$H_i = \boldsymbol{k}_S^T * (\boldsymbol{k}_S * M_i)$$
$$\boldsymbol{k}_S = (1 \quad \cdots \quad 1)$$

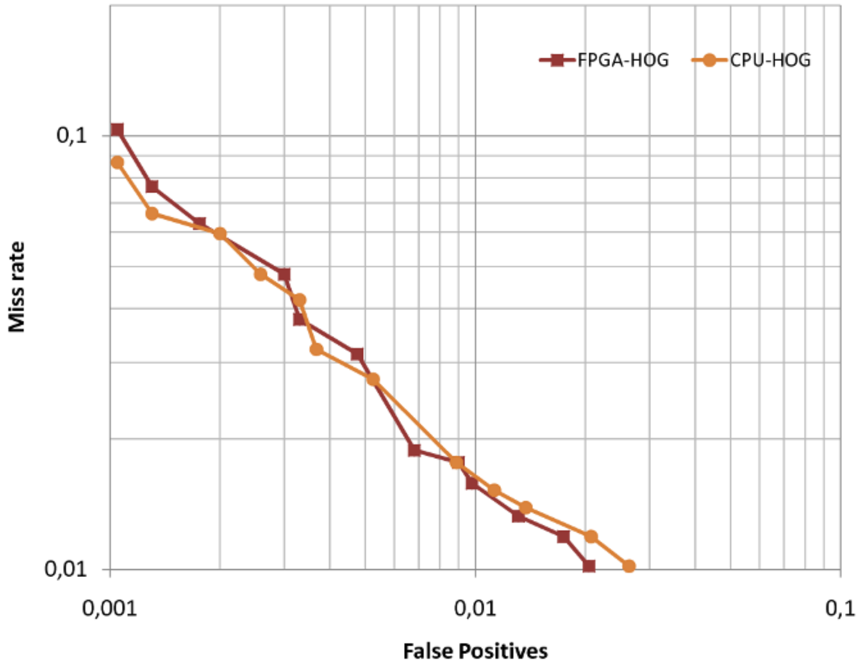


S. Bauer, S. Köhler, K. Doll, U. Brunsmann

HOG Classification Performance

- Evaluation on INRIA benchmark[4]

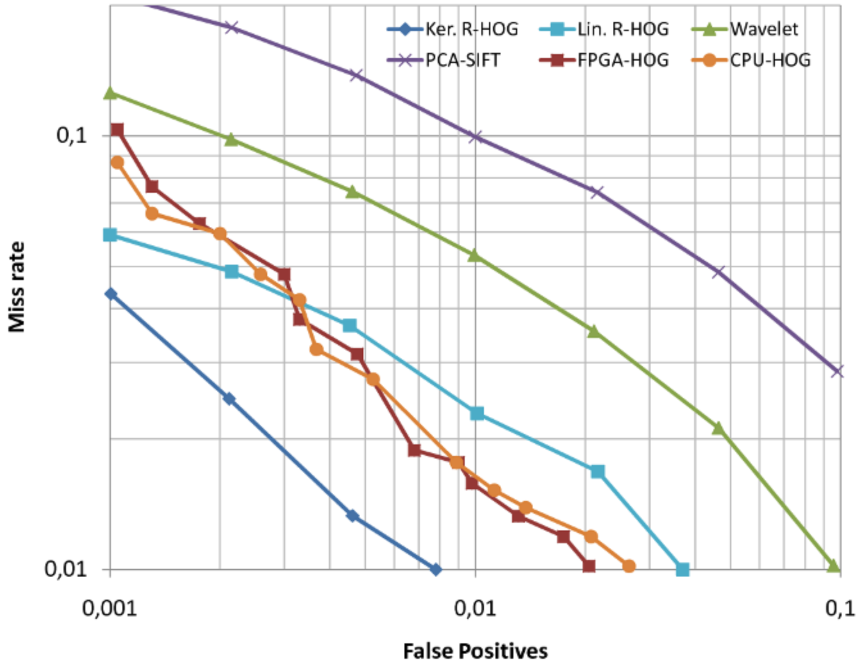
  Detection error tradeoff (DET) curve:



[4] N. Dalal, B. Triggs: Histograms of oriented gradients for human detection. CVPR, 2005.

HOG Classification Performance

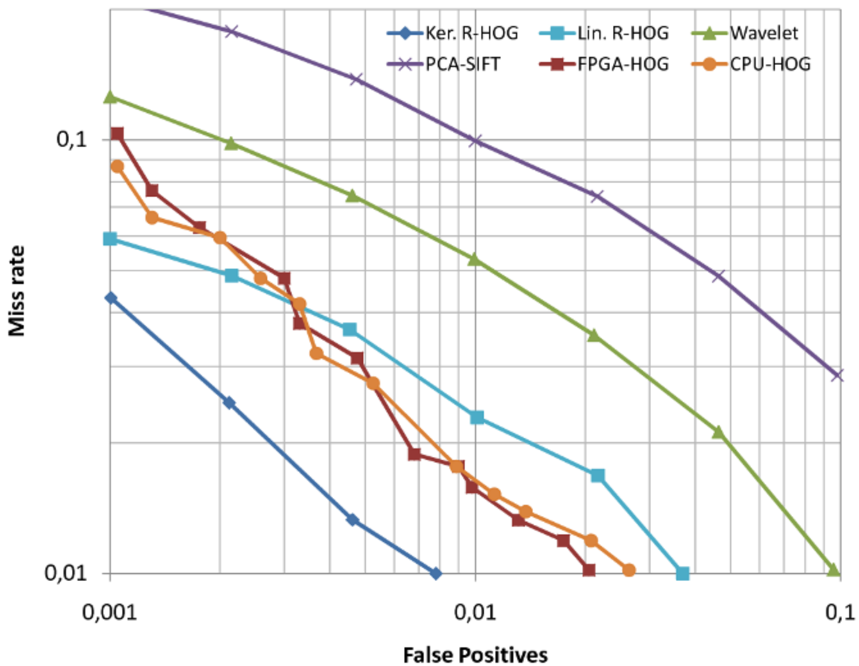
- Evaluation on INRIA benchmark[4]
  Detection error tradeoff (DET) curve:



[4] N. Dalal, B. Triggs: Histograms of oriented gradients for human detection. CVPR, 2005.

HOG Classification Performance

- Evaluation on INRIA benchmark[4]
  Detection error tradeoff (DET) curve:



FPGA-HOG Latency

HOG computation step	Latency in [μs]	
Image buffer	26.2	8.4%
Scaling	26.0	8.3%
Gradients, magnitudes, orientations	52.2	16.8%
Histograms	207.2	66.5%
Total	311.6	

Throughput: full 30 fps (1600x1200 px)

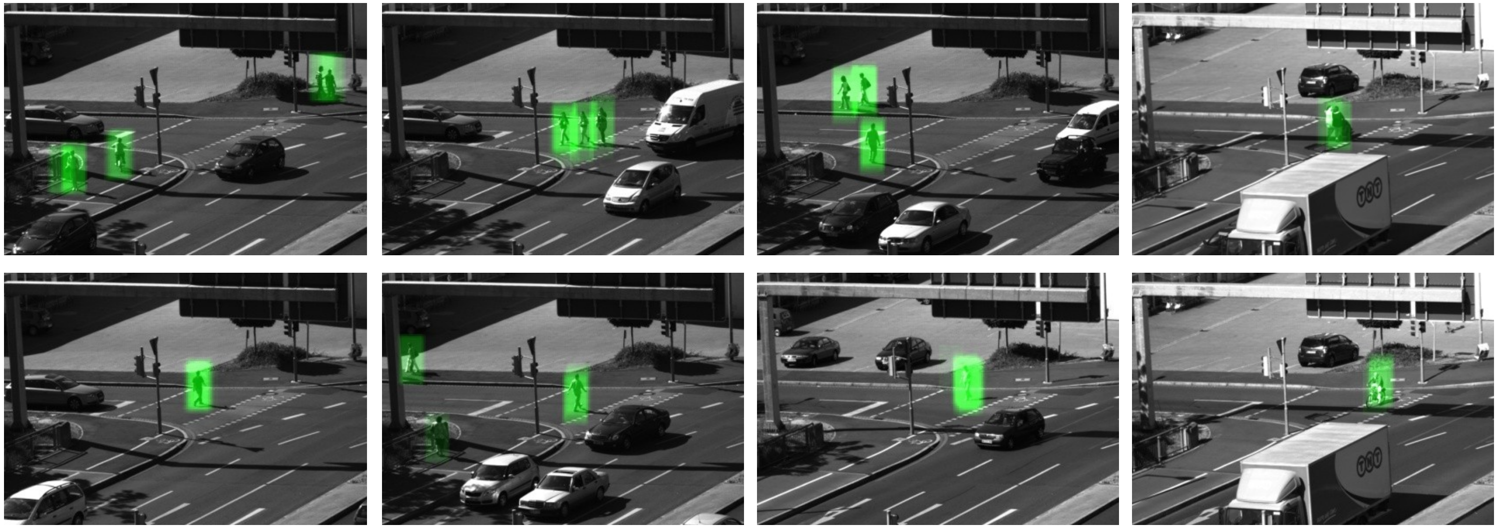
Resources

Type of Resource	Usage	
4-input LUTs	28,616	46%
Internal Block RAM (18 kbit)	100	61%
Embedded Multipliers (18 × 18)	18	18%

Spartan 3 XC3S 4000, processing 800x600 px

[4] N. Dalal, B. Triggs: Histograms of oriented gradients for human detection. CVPR, 2005.

- Detection performance (intersection assistance)
  - Evaluation on 1000 frames, manually labeled ground truth
  - Hypothesis generation: Detection rate: > 80%
  - HOG/kernel SVM verification: Detection rate: 95.4%
    False alarm rate: 0.1%



- System performance (intersection assistance)
  - Framerate: > 10 fps (evaluation of 10 hypotheses)

Summary

- FPGA-GPU architecture for pedestrian detection
- HOG descriptor on low-cost FPGA
- Real-time system for crossroad assistance (> 10 fps)

Outlook

- Outsource further parts onto FPGA
  - Multi-spectral hypothesis generation
  - HOG normalization
- Incorporate object tracking
- Application to bicyclists, motorcyclists

Outlook

Summary

- FPGA-GPU architecture for pedestrian detection
- HOG descriptor on low-cost FPGA
- Real-time system for crossroad assistance (> 10 fps)

Outlook

- Outsource further parts onto FPGA
  - Multi-spectral hypothesis generation
  - HOG normalization
- Incorporate object tracking
- Application to bicyclists, motorcyclists