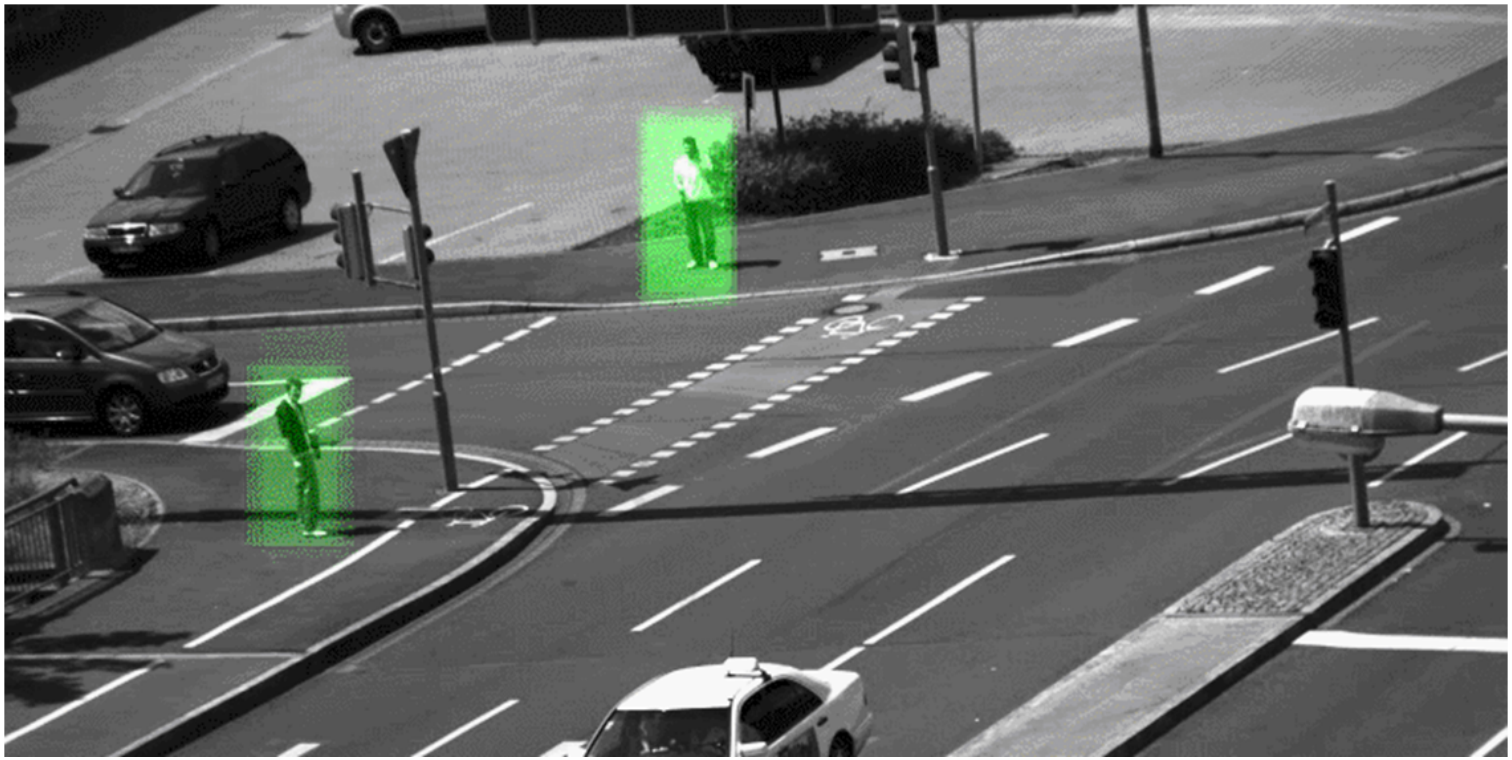


MPC Workshop
Karlsruhe
10/7/2009

FPGA Implementation of a HOG-based **Pedestrian Recognition System**

Sebastian Bauer | sebastian.bauer@fh-aschaffenburg.de

Laboratory for Pattern Recognition and Computational Intelligence
Prof. Dr. Ulrich Brunsmann

FH Aschaffenburg
University of Applied Sciences

Content

- Motivation
- Method
- Implementation
- Evaluation
- Summary and Outlook

Motivation

- Problem Scope
  - 1,200,000 road traffic deaths per year [WHO/Worldbank 2004]
  - Majority among vulnerable road users (VRU)
  - **No advanced driver assistance system (ADAS) available for daylight pedestrian recognition**

Motivation | Method | Implementation | Evaluation | Summary

- Problem Scope
  - 1,200,000 road traffic deaths per year [WHO/Worldbank 2004]
  - Majority among vulnerable road users (VRU)
  - **No advanced driver assistance system (ADAS) available for daylight pedestrian recognition**

- State-of-the-Art
  - HOG detection system yields best performance [Dollár 2009],[Enzweiler 2009]
  - Challenge: CPU computation time: > 10s (VGA)

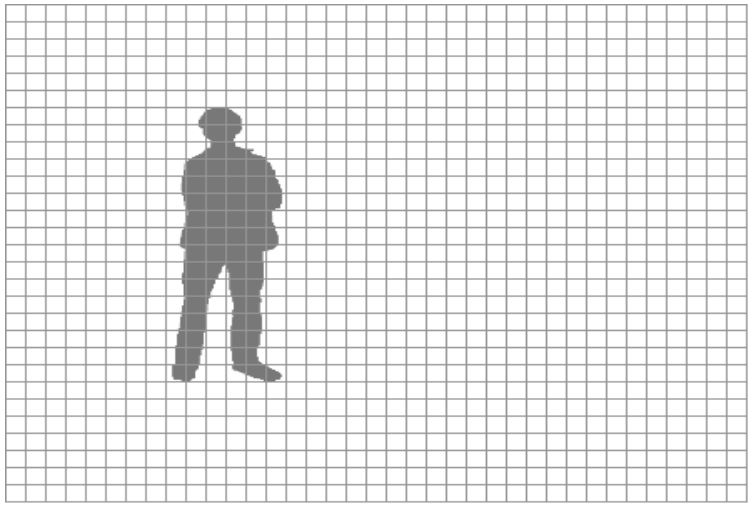
Motivation

- Problem Scope
  - 1,200,000 road traffic deaths per year [WHO/Worldbank 2004]
  - Majority among vulnerable road users (VRU)
  - **No advanced driver assistance system (ADAS) available for daylight pedestrian recognition**

- State-of-the-Art
  - HOG detection system yields best performance [Dollár 2009],[Enzweiler 2009]
  - Challenge: CPU computation time: > 10s (VGA)
  - **Goal: Real-time HOG system with FPGA**

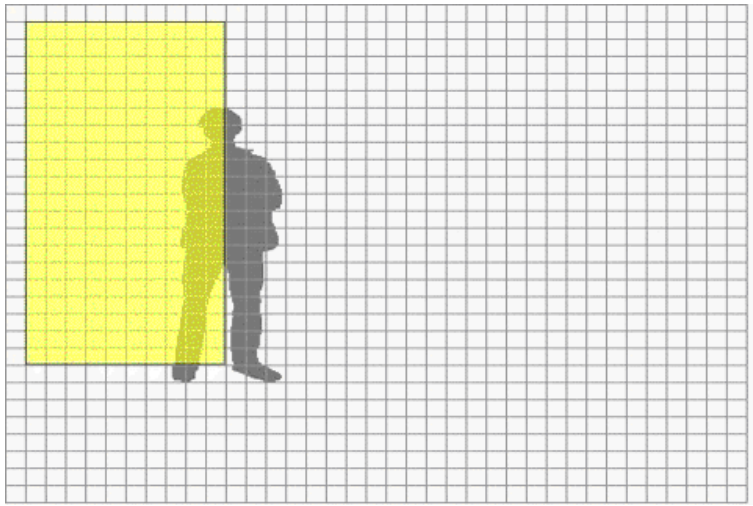
- HOG Detection System
  - **Sliding Detection Window**
    - For each window:

1. Generate **Feature Set / „Descriptor“** (HOG)
2. **Classify** pedestrian / non-pedestrian



- HOG Detection System
  - **Sliding Detection Window**
    - For each window:

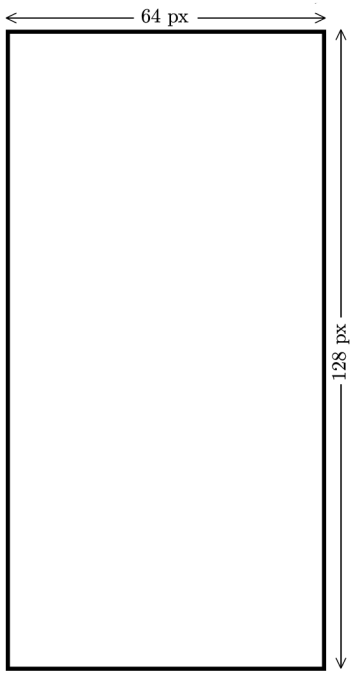
1. Generate **Feature Set / „Descriptor“** (HOG)
2. **Classify** pedestrian / non-pedestrian



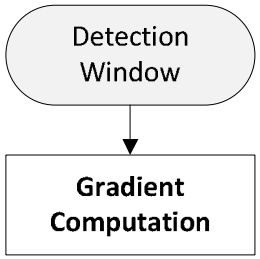
- HOG Descriptor – Histograms of Oriented Gradients [Dalal 2005]

Detection Window

- **Detection Window**



HOG Descriptor – Histograms of Oriented Gradients [Dalal 2005]



- **Gradient Computation**
  - Point derivatives

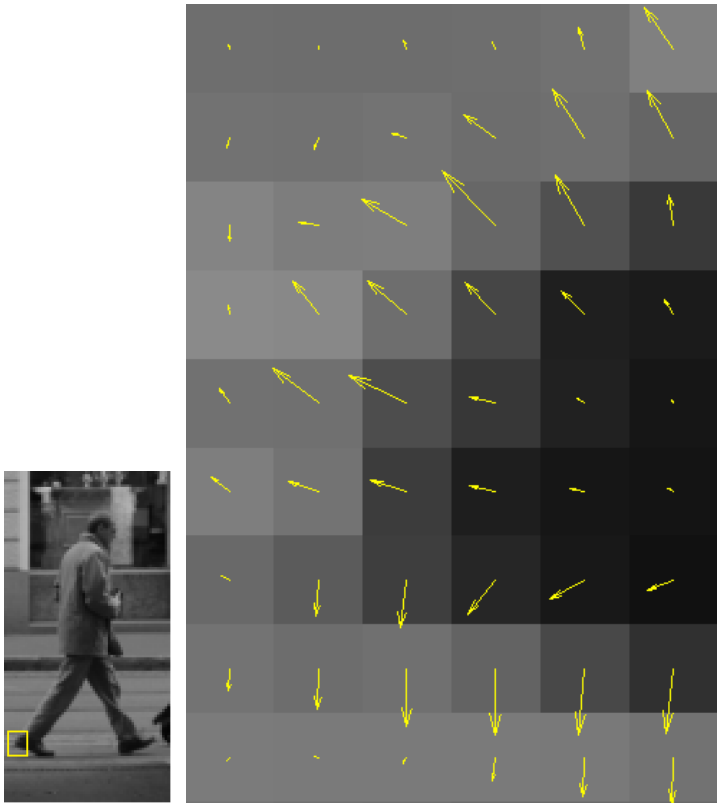
$$G_x = M_x * I \qquad M_x = [-1 \;\; 0 \;\; 1]$$
$$G_y = M_y * I \qquad M_y = [-1 \;\; 0 \;\; 1]^T$$

  - Magnitude

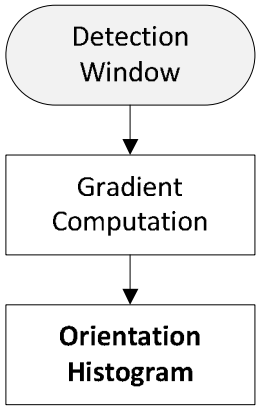
$$|G(x,y)| = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$$

  - Orientation

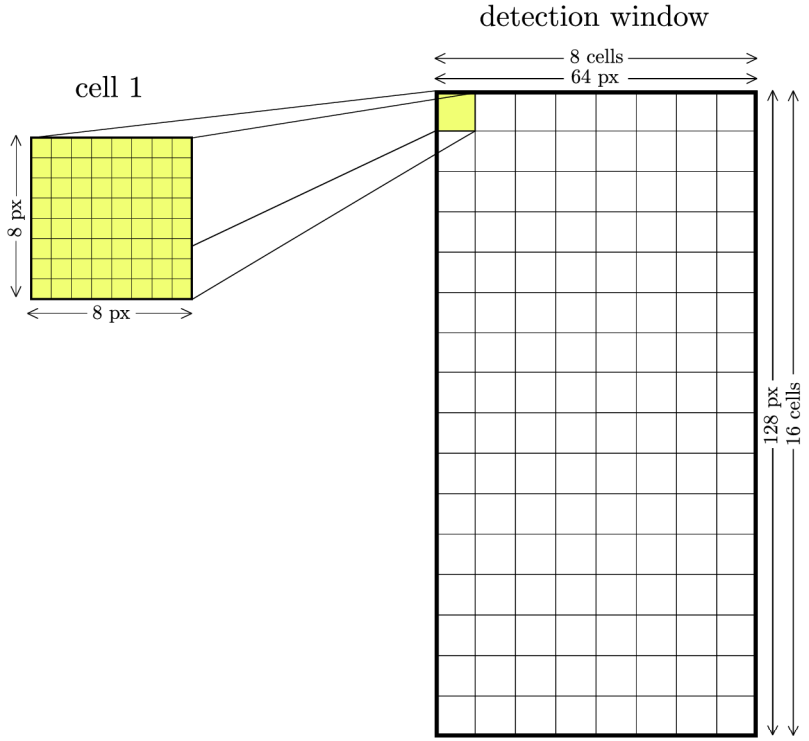
$$\tan\big(\phi(x,y)\big) = \frac{G_y(x,y)}{G_x(x,y)}$$



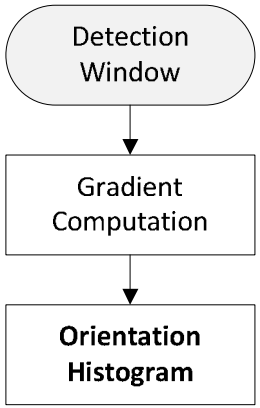
- HOG Descriptor



- **Orientation Histogram**
  - Detection Window subdivided into **Cells**

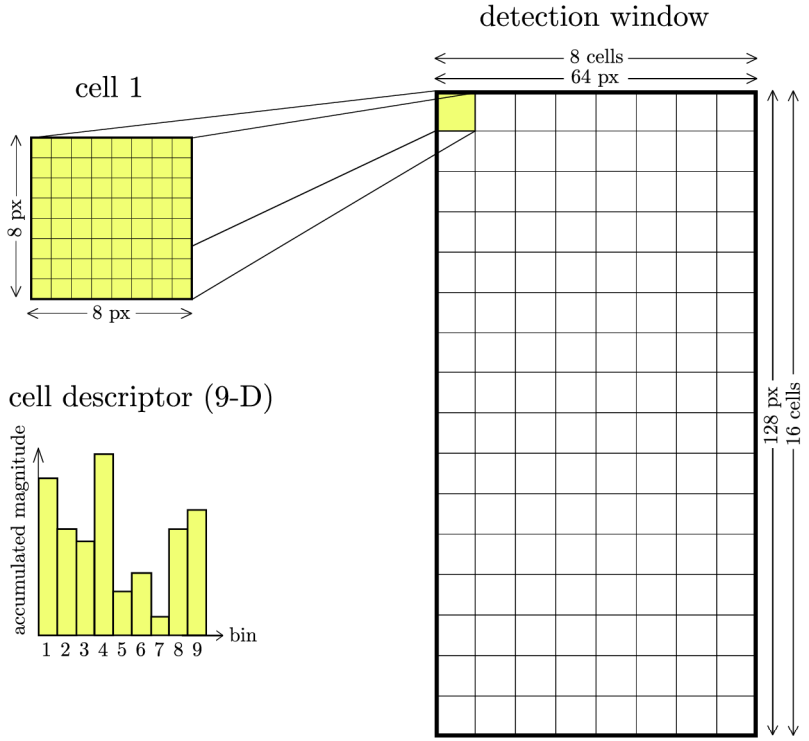


- HOG Descriptor

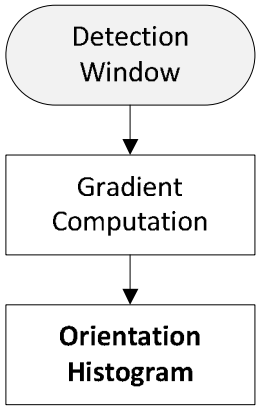


- **Orientation Histogram**
  - Detection Window subdivided into **Cells**
  - For each Cell:

    Histogram Generation

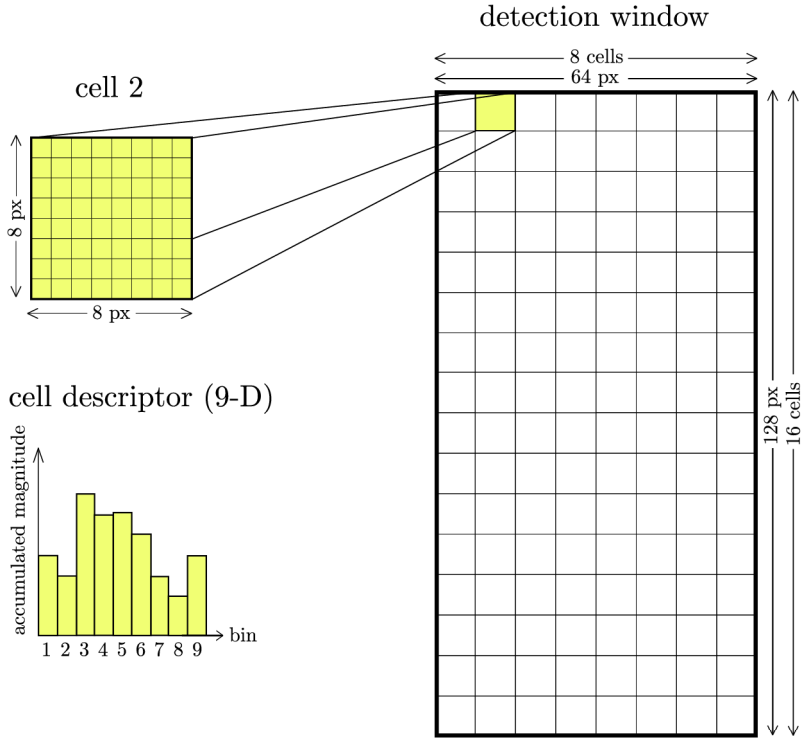


- HOG Descriptor

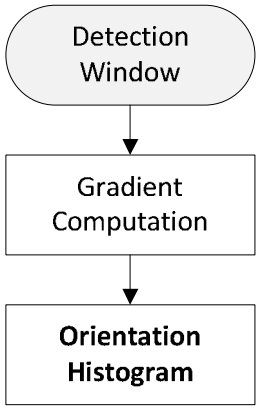


- **Orientation Histogram**
  - Detection Window subdivided into **Cells**
  - For each Cell:

    Histogram Generation

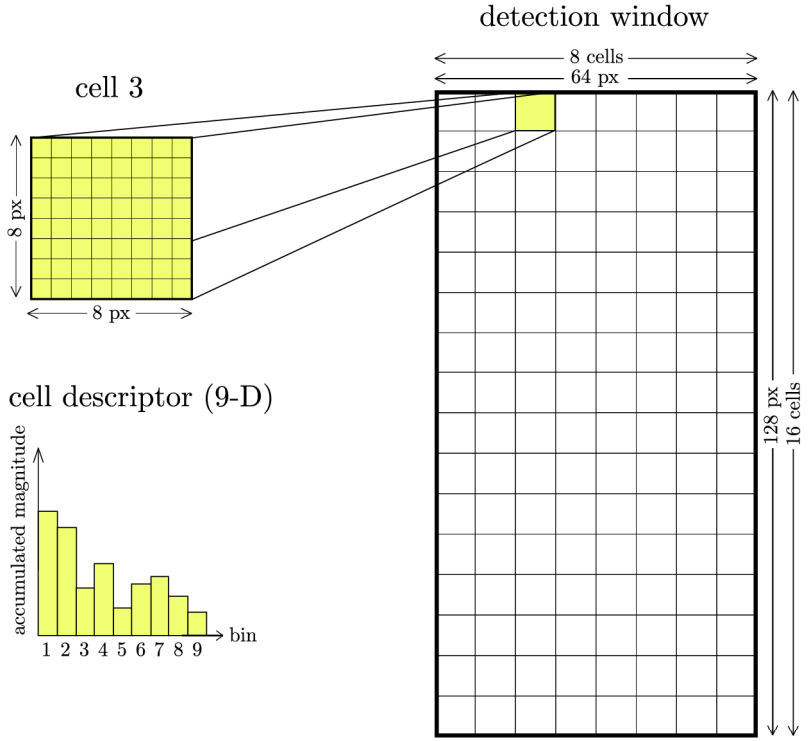


- HOG Descriptor

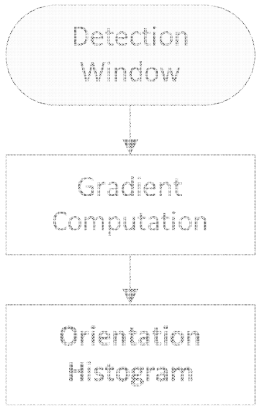


- **Orientation Histogram**
  - Detection Window subdivided into **Cells**
  - For each Cell:

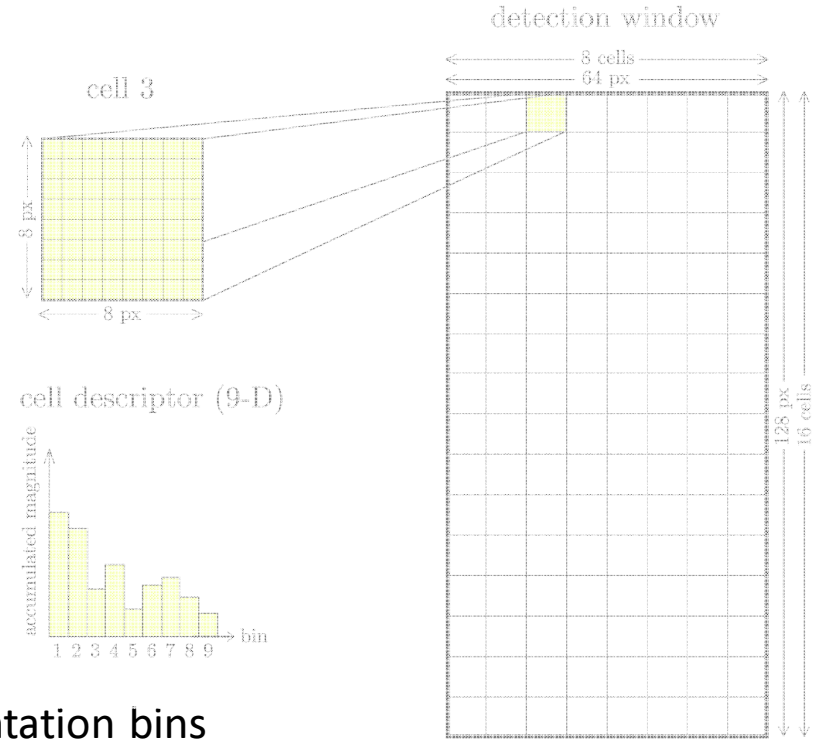
    Histogram Generation



- HOG Descriptor



- Orientation Histogram
  - Detection Window subdivided into Cells
  - For each Cell: Histogram Generation



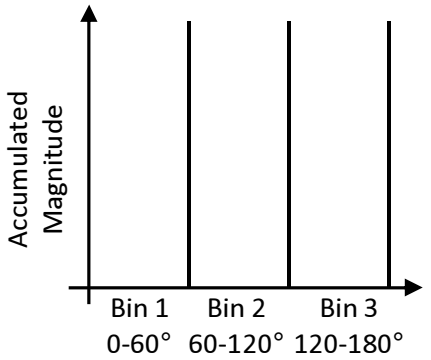
- **Example:** 4x4 px Cell, 3 orientation bins

|G|

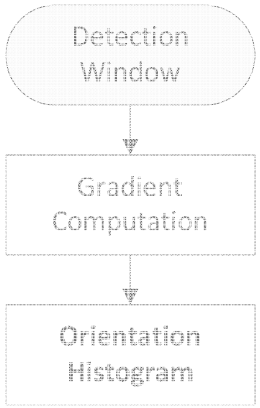
12	42	152	232
18	91	64	31
27	48	219	102
15	43	147	160

ϕ

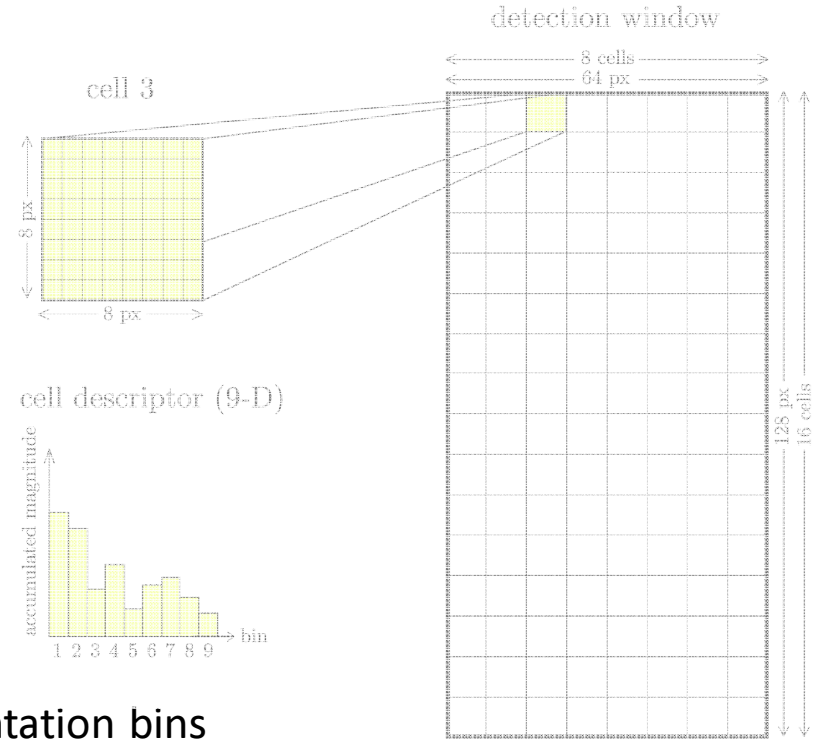
47°	21°	78°	170°
98°	94°	5°	140°
27°	77°	30°	128°
4°	11°	22°	101°



HOG Descriptor



- Orientation Histogram
  - Detection Window subdivided into Cells
  - For each Cell: Histogram Generation



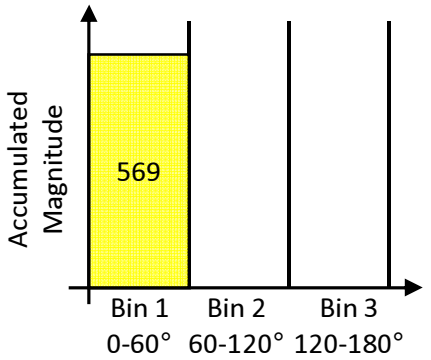
- **Example:** 4x4 px Cell, 3 orientation bins

|G|

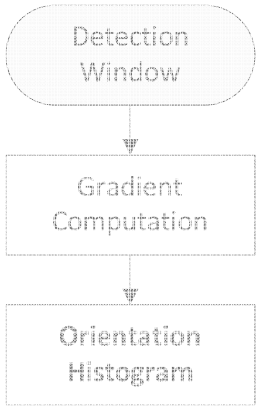
12	42	152	232
18	91	64	31
27	48	219	102
15	43	147	160

ϕ

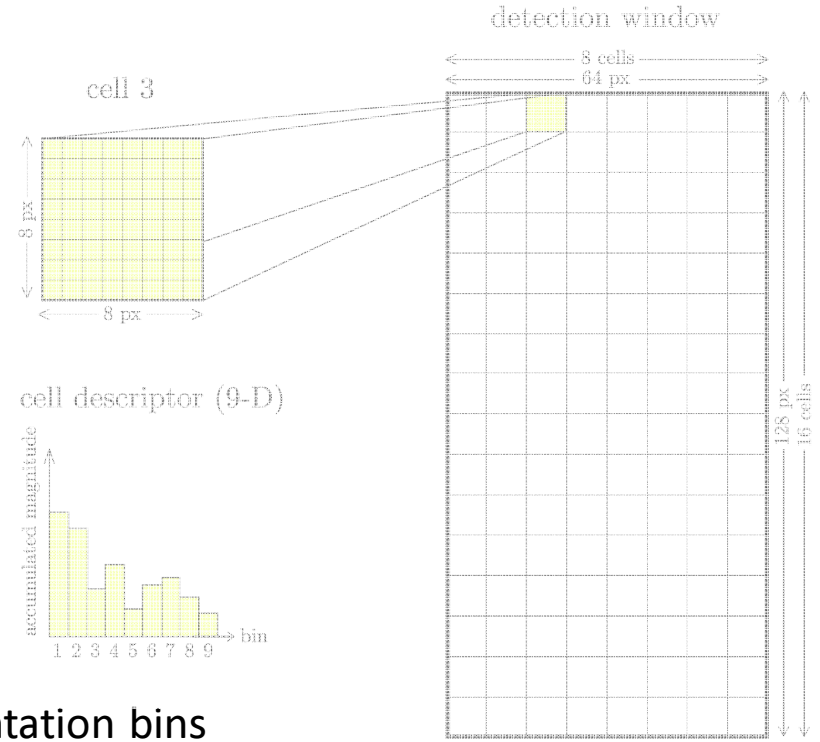
47°	21°	78°	170°
98°	94°	5°	140°
27°	77°	30°	128°
4°	11°	22°	101°



- HOG Descriptor



- Orientation Histogram
  - Detection Window subdivided into Cells
  - For each Cell: Histogram Generation



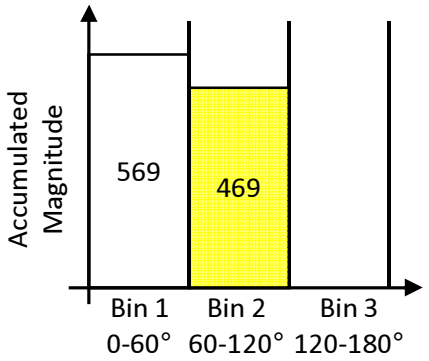
- **Example:** 4x4 px Cell, 3 orientation bins

|G|

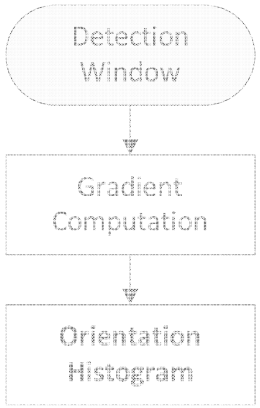
12	42	152	232
18	91	64	31
27	48	219	102
15	43	147	160

ϕ

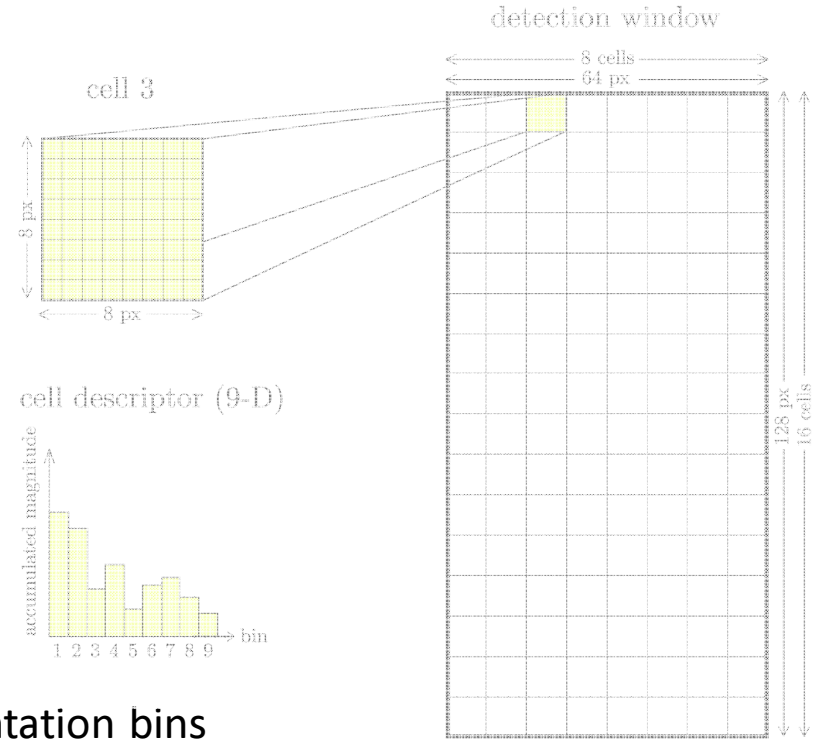
47°	21°	78°	170°
98°	94°	5°	140°
27°	77°	30°	128°
4°	11°	22°	101°



HOG Descriptor



- Orientation Histogram
  - Detection Window subdivided into Cells
  - For each Cell: Histogram Generation



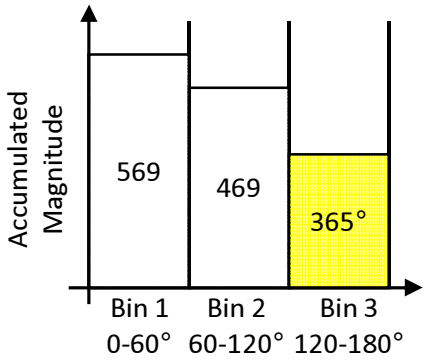
- **Example:** 4x4 px Cell, 3 orientation bins

|G|

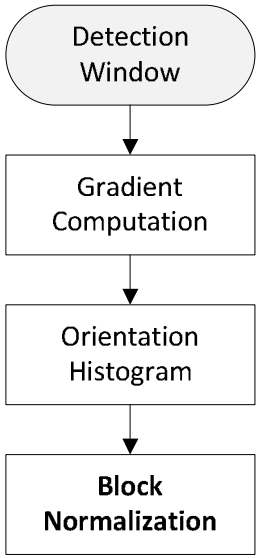
12	42	152	232
18	91	64	31
27	48	219	102
15	43	147	160

ϕ

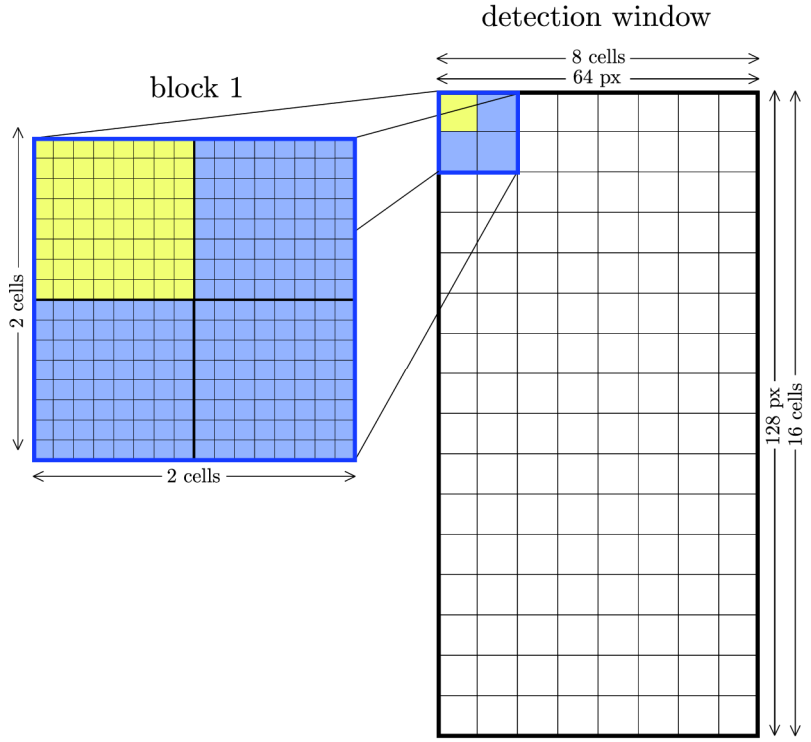
47°	21°	78°	170°
98°	94°	5°	140°
27°	77°	30°	128°
4°	11°	22°	101°



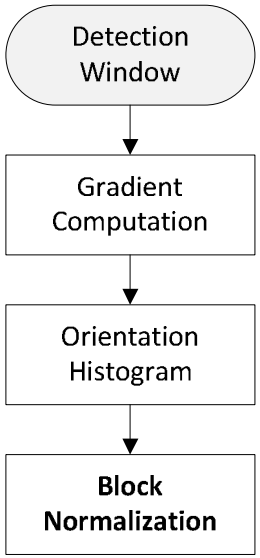
- HOG Descriptor



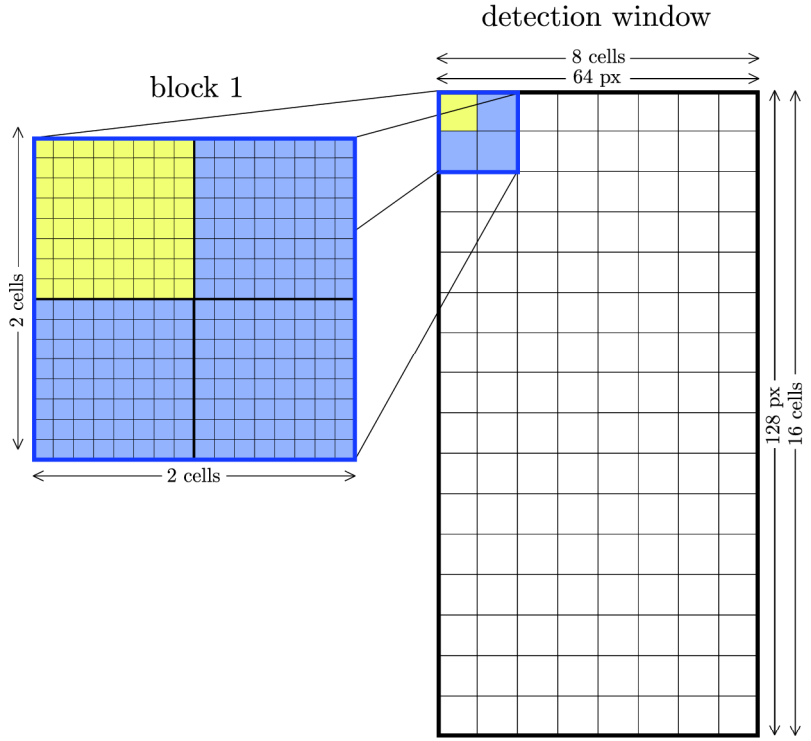
- **Block Normalization**
  - Block: 2x2 Cells

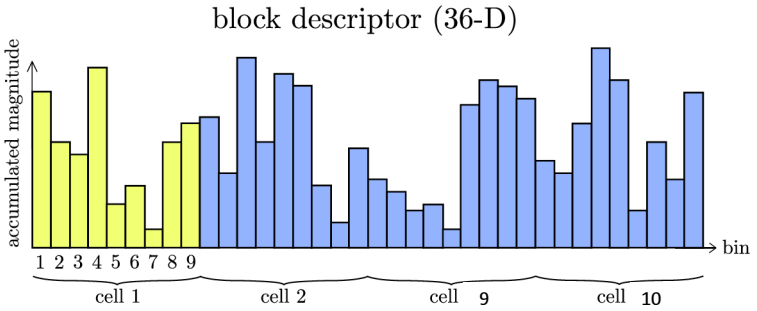


- HOG Descriptor

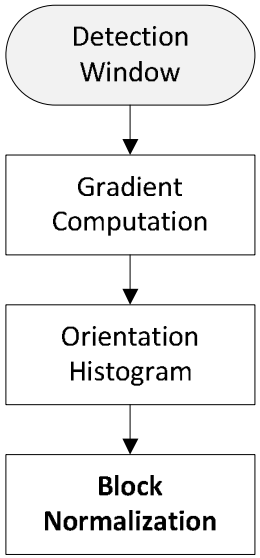


- **Block Normalization**
  - Block: 2x2 Cells
  - For each Block:
    Concatenation of Cell histograms





- HOG Descriptor

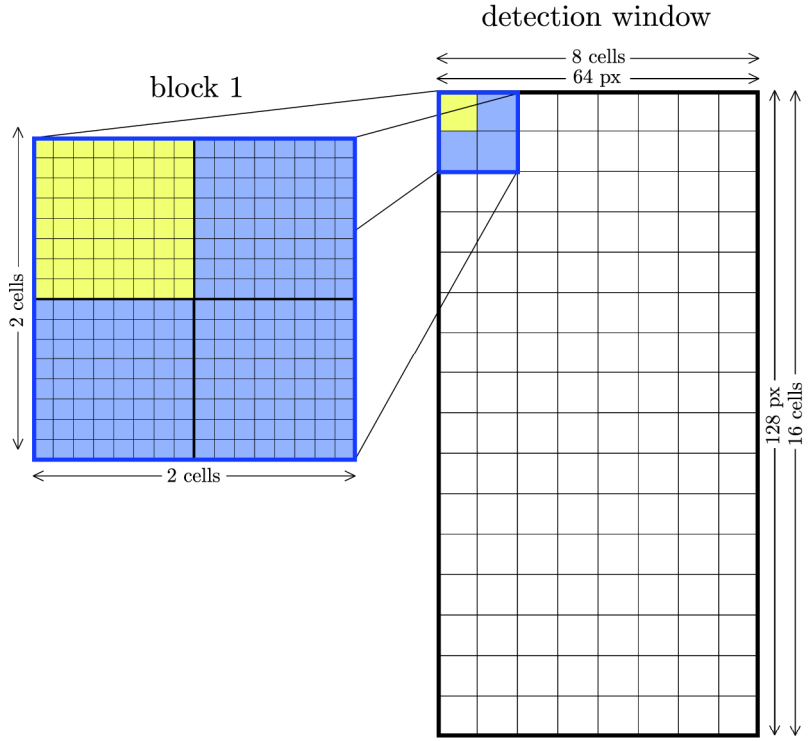


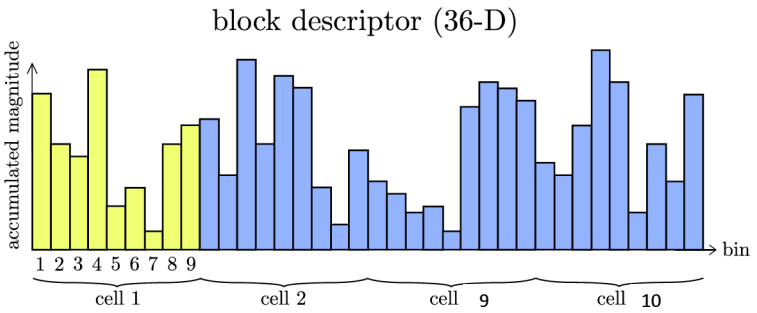
- **Block Normalization**
  - Block: 2x2 Cells
  - For each Block:

    Concatenation of Cell histograms

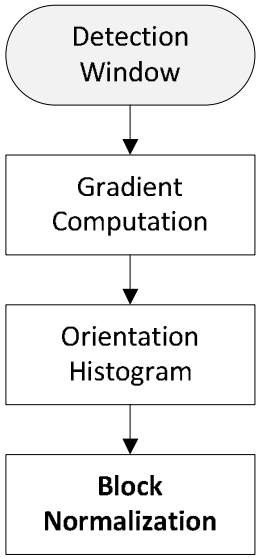
    Normalization using L2-Norm

$$v \longrightarrow \frac{v}{\sqrt{\|v\|_2^2 + \epsilon^2}}$$





- HOG Descriptor

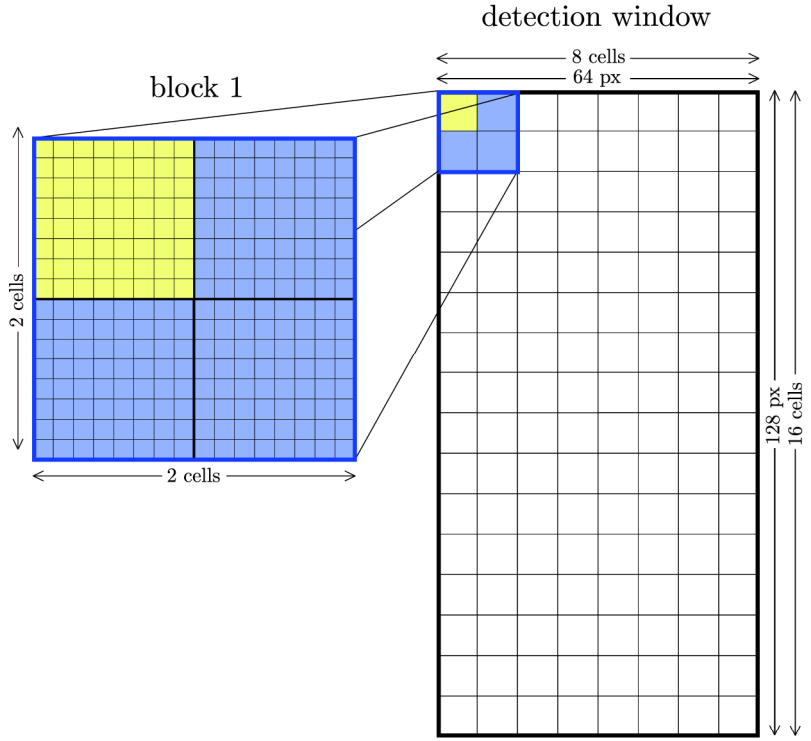


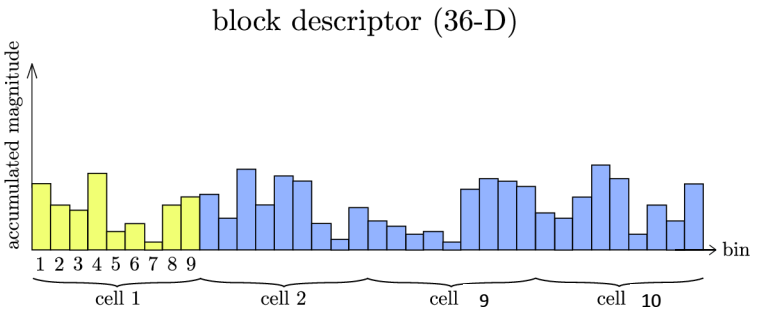
- **Block Normalization**
  - Block: 2x2 Cells
  - For each Block:

    Concatenation of Cell histograms

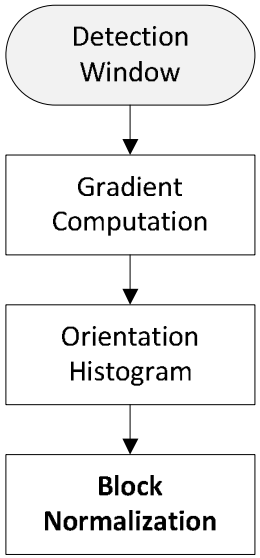
    Normalization using L2-Norm

$$v \longrightarrow \frac{v}{\sqrt{\|v\|_2^2 + \epsilon^2}}$$





- HOG Descriptor

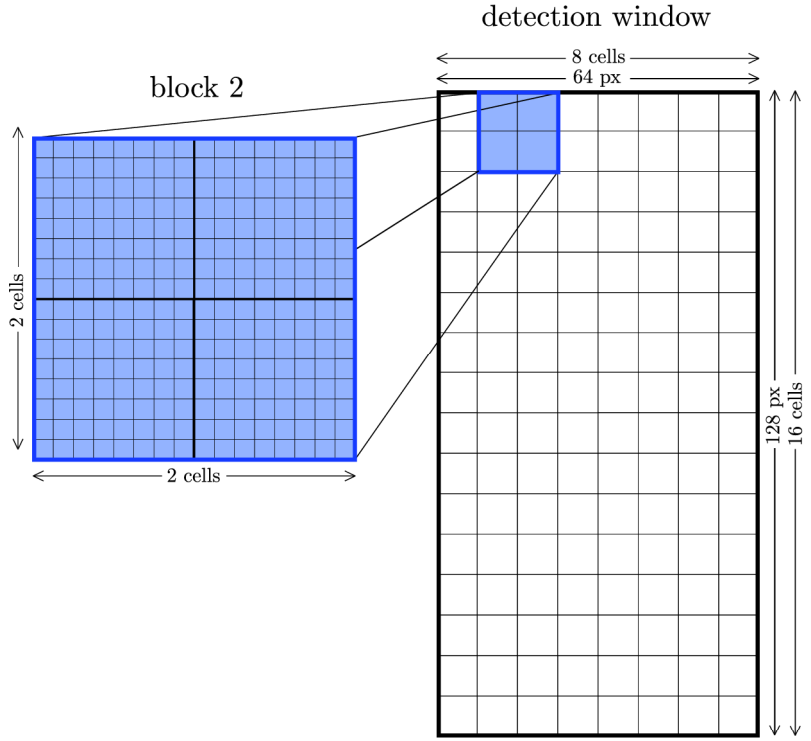


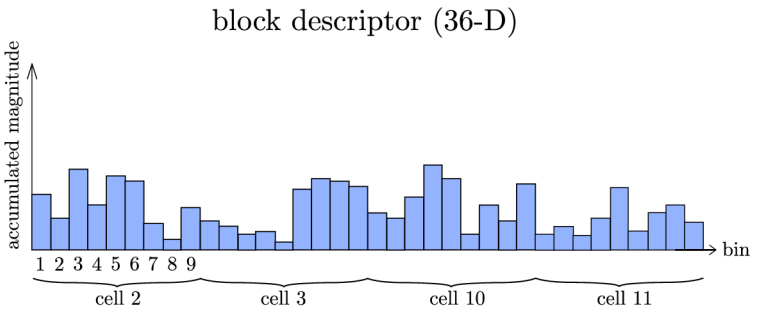
- **Block Normalization**
  - Block: 2x2 Cells
  - For each Block:

    Concatenation of Cell histograms

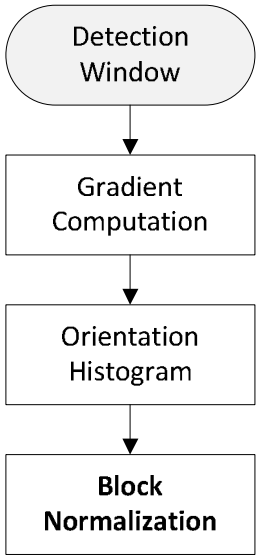
    Normalization using L2-Norm

$$v \longrightarrow \frac{v}{\sqrt{\|v\|_2^2 + \epsilon^2}}$$





- HOG Descriptor

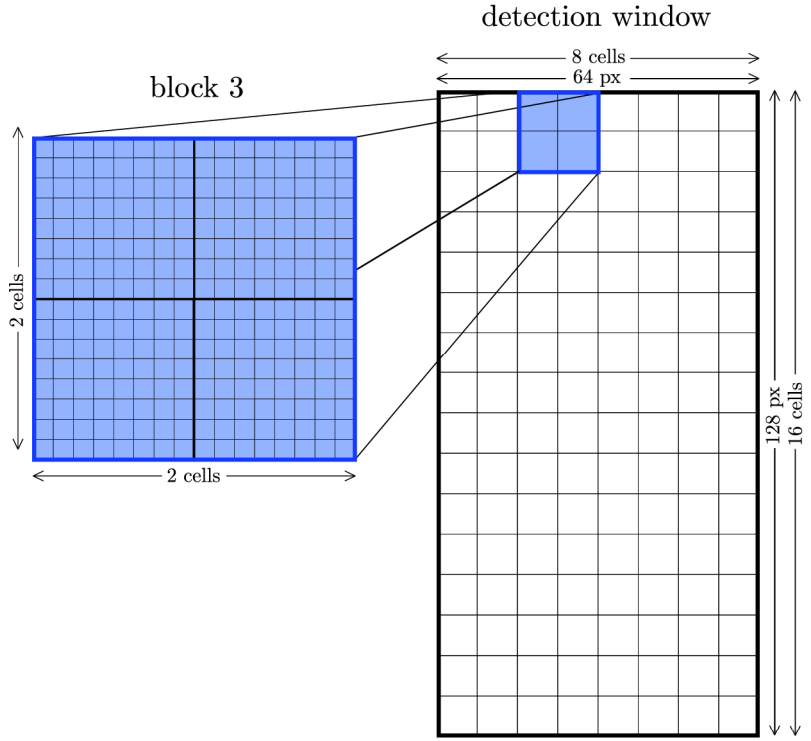


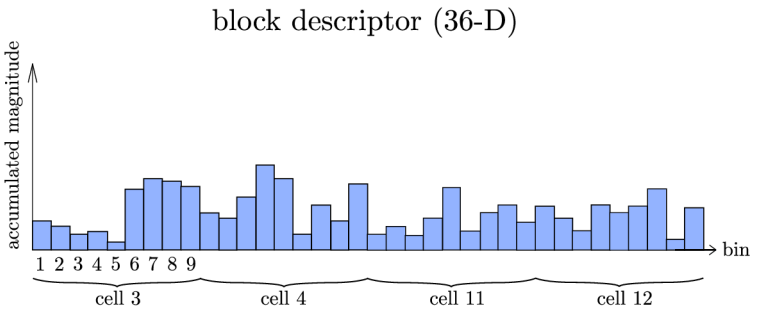
- **Block Normalization**
  - Block: 2x2 Cells
  - For each Block:

    Concatenation of Cell histograms

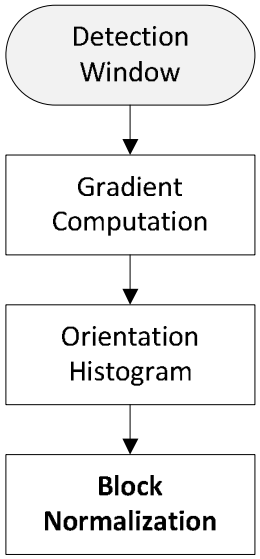
    Normalization using L2-Norm

$$v \longrightarrow \frac{v}{\sqrt{\|v\|_2^2 + \epsilon^2}}$$





- HOG Descriptor

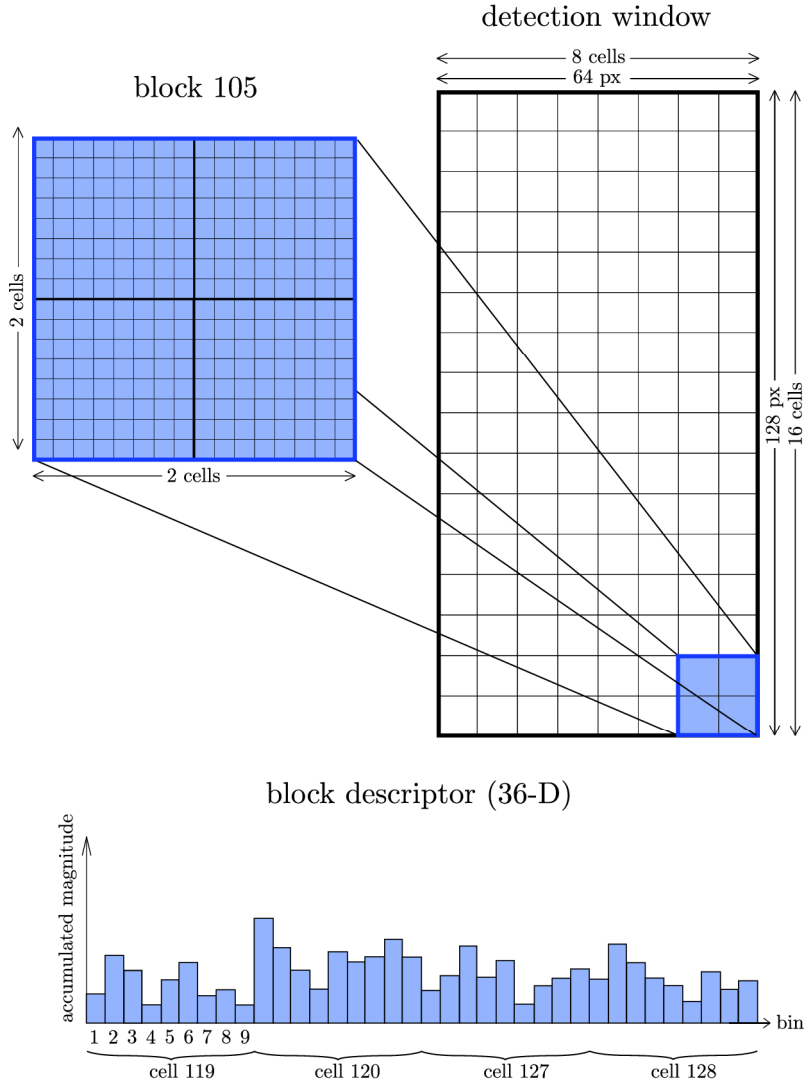


- **Block Normalization**
  - Block: 2x2 Cells
  - For each Block:

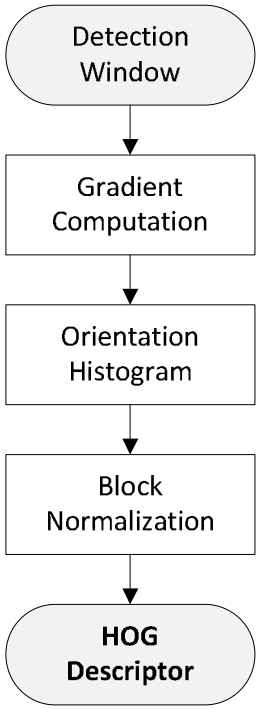
    Concatenation of Cell histograms

    Normalization using L2-Norm

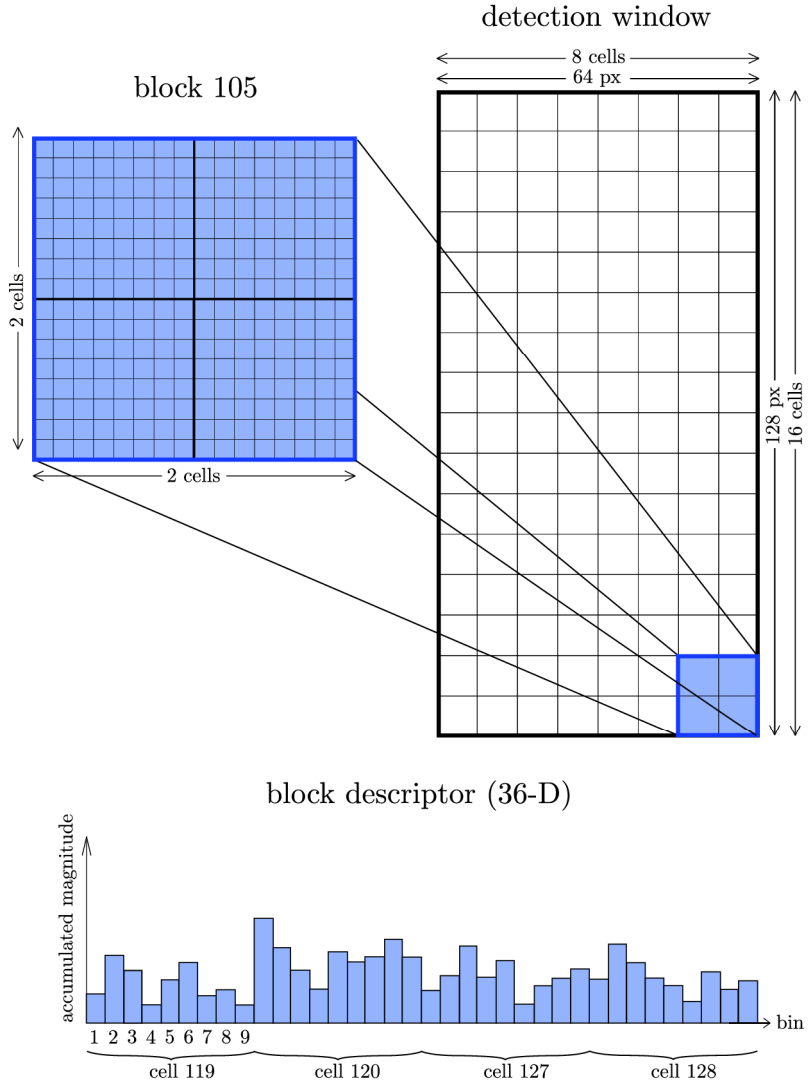
$$v \longrightarrow \frac{v}{\sqrt{\|v\|_2^2 + \epsilon^2}}$$



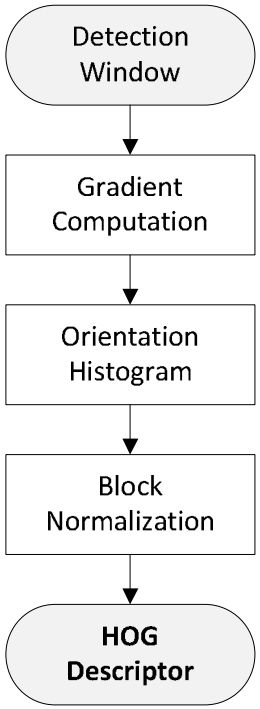
- HOG Descriptor



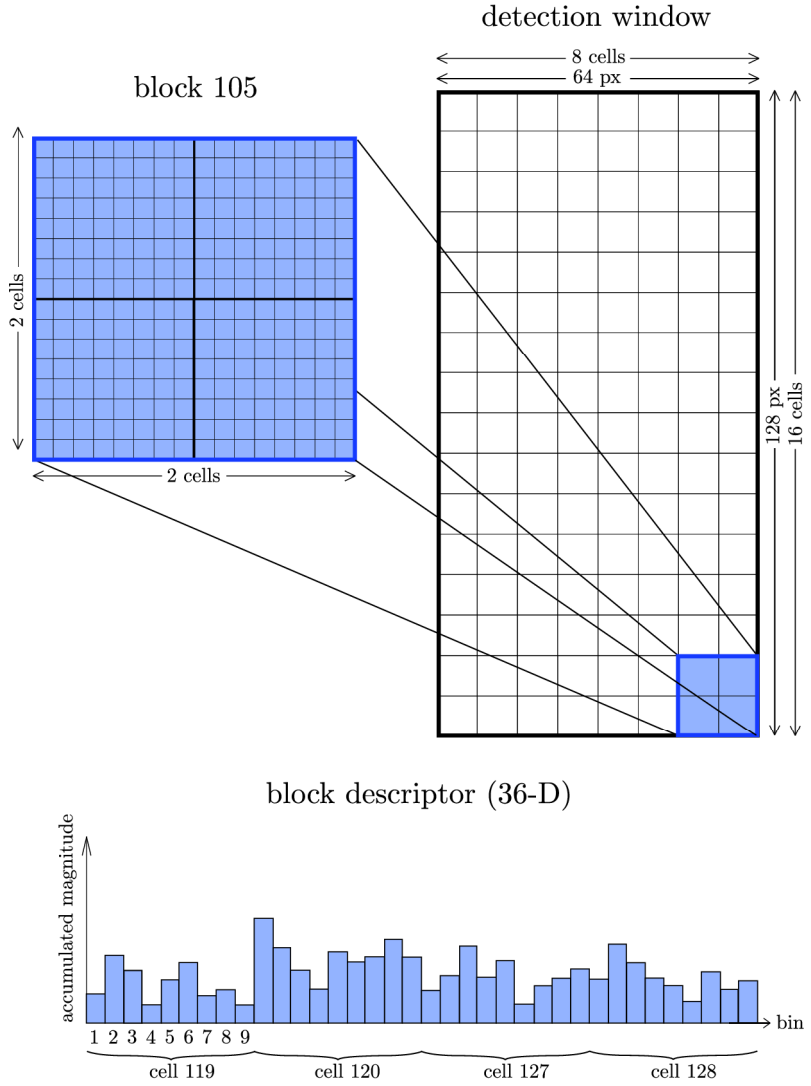
- **HOG Descriptor**



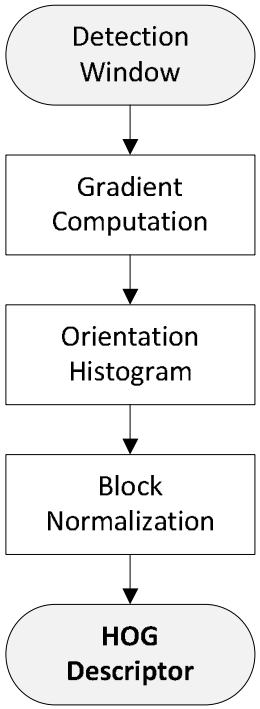
- HOG Descriptor



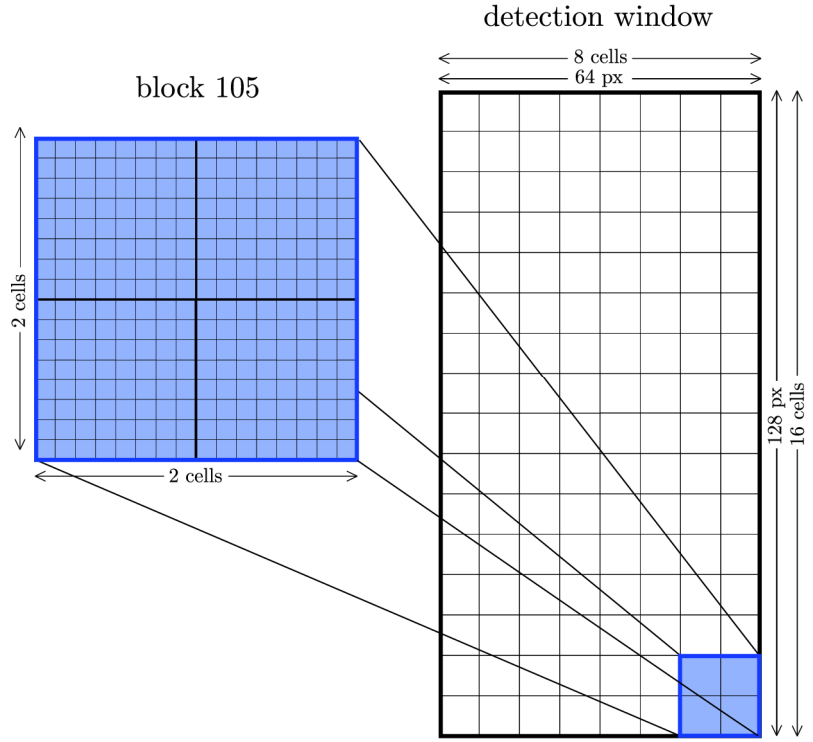
- **HOG Descriptor**
  - Concatenation of normalized Blocks
  - Feature Vector with 3780 dimensions

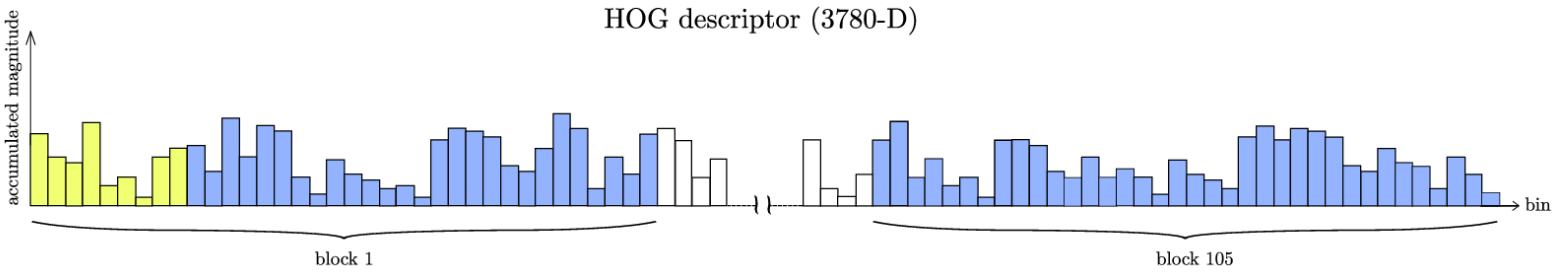


- HOG Descriptor

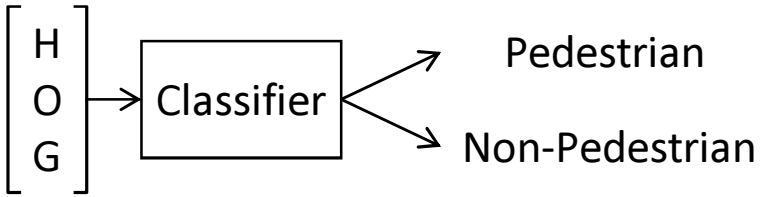


- **HOG Descriptor**
  - Concatenation of normalized Blocks
  - Feature Vector with 3780 dimensions



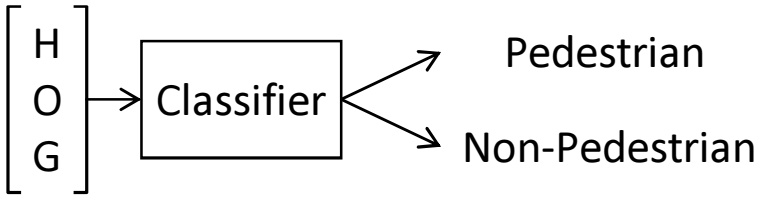


- Classification

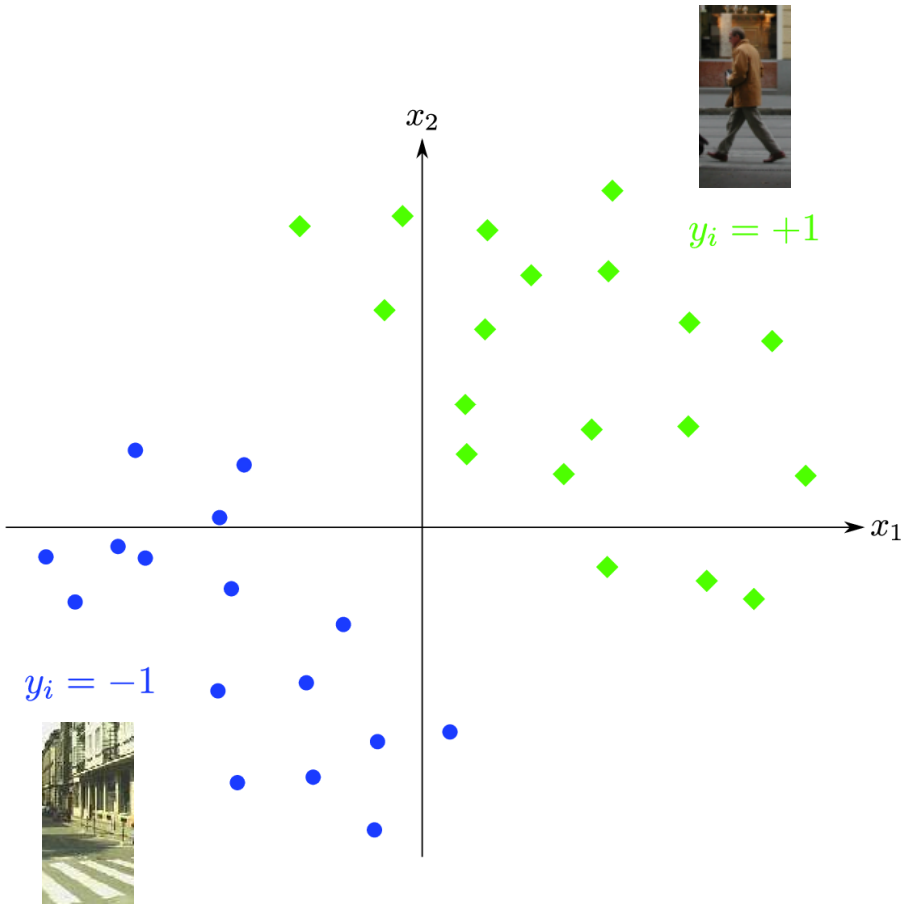


  - **Machine Learning**
    - Learning from Examples (I/O pairs)
    - Established Pedestrian Datasets [INRIA]

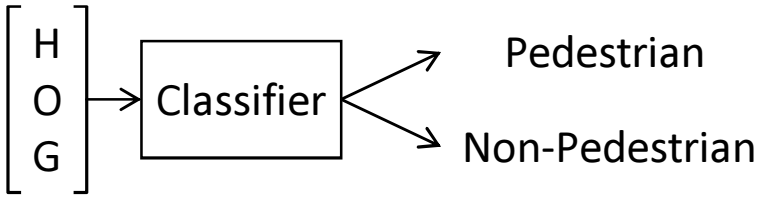
- Classification



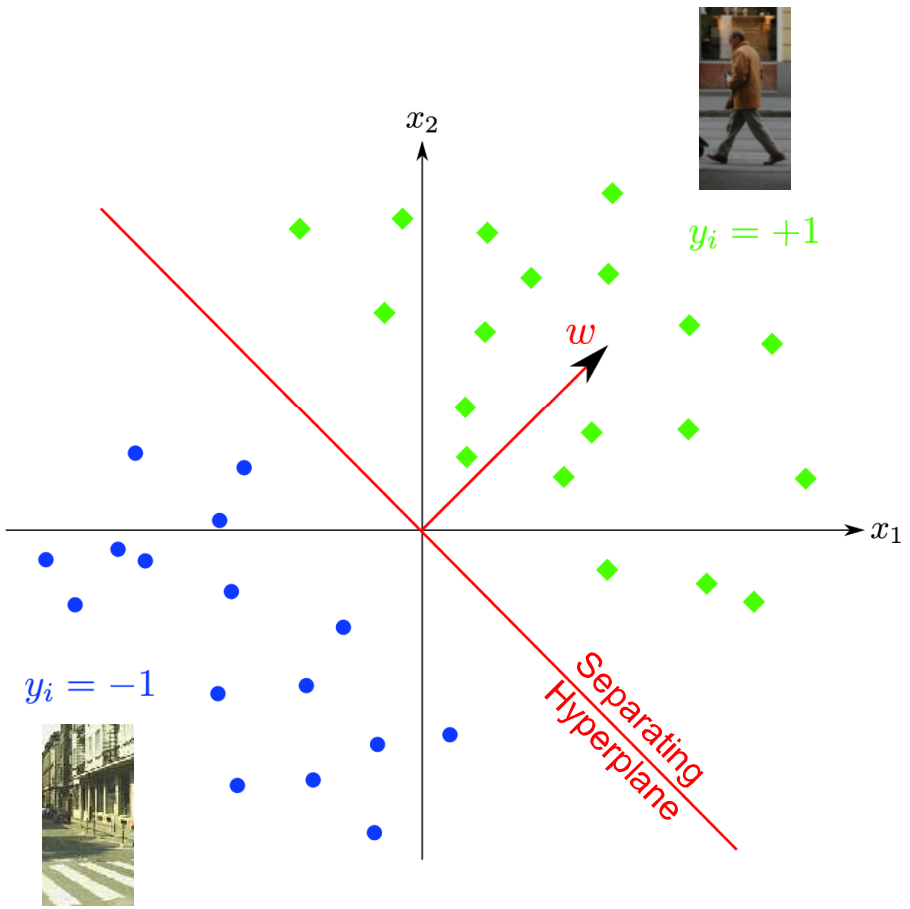
  - **Machine Learning**
    - Learning from Examples (I/O pairs)
    - Established Pedestrian Datasets [INRIA]
  - **Support Vector Machines (SVM)**
    - Linear Classifier
    - Decision function:
      $y = f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w} \cdot \mathbf{x})$
    - **w** learned by training



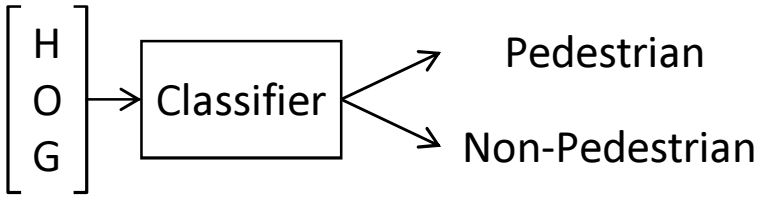
- Classification



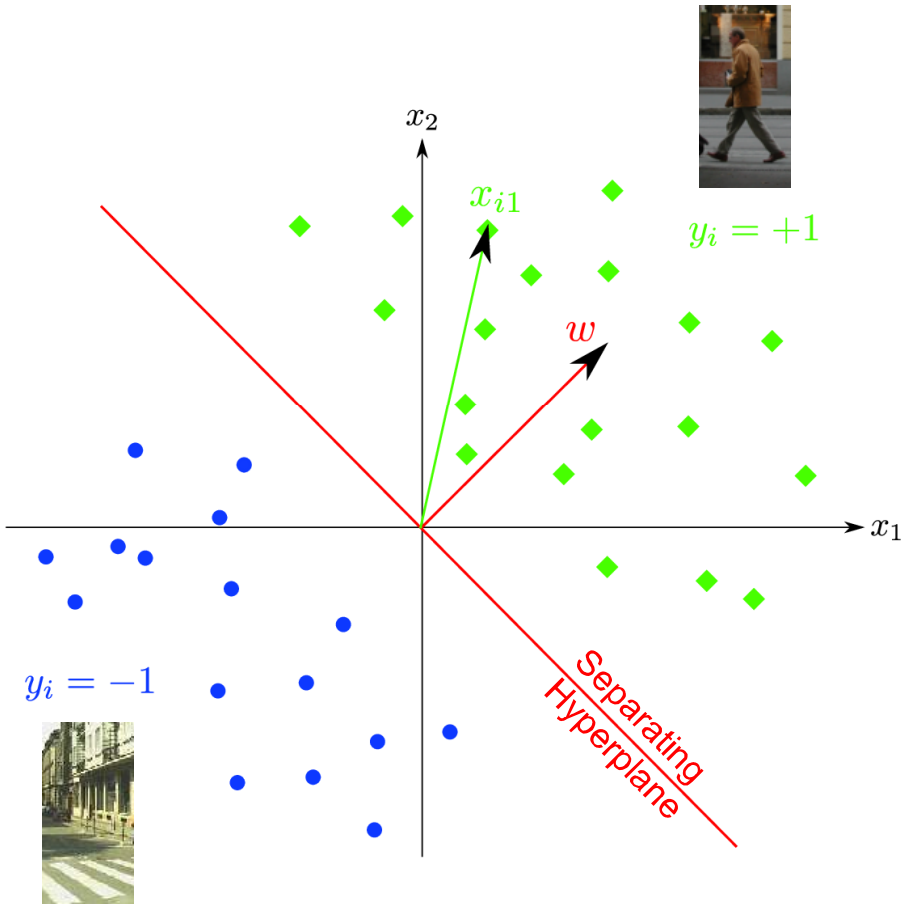
- **Machine Learning**
  - Learning from Examples (I/O pairs)
  - Established Pedestrian Datasets [INRIA]
  - **Support Vector Machines (SVM)**
    - Linear Classifier
    - Decision function:
      $y = f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w} \cdot \mathbf{x})$
    - **w** learned by training



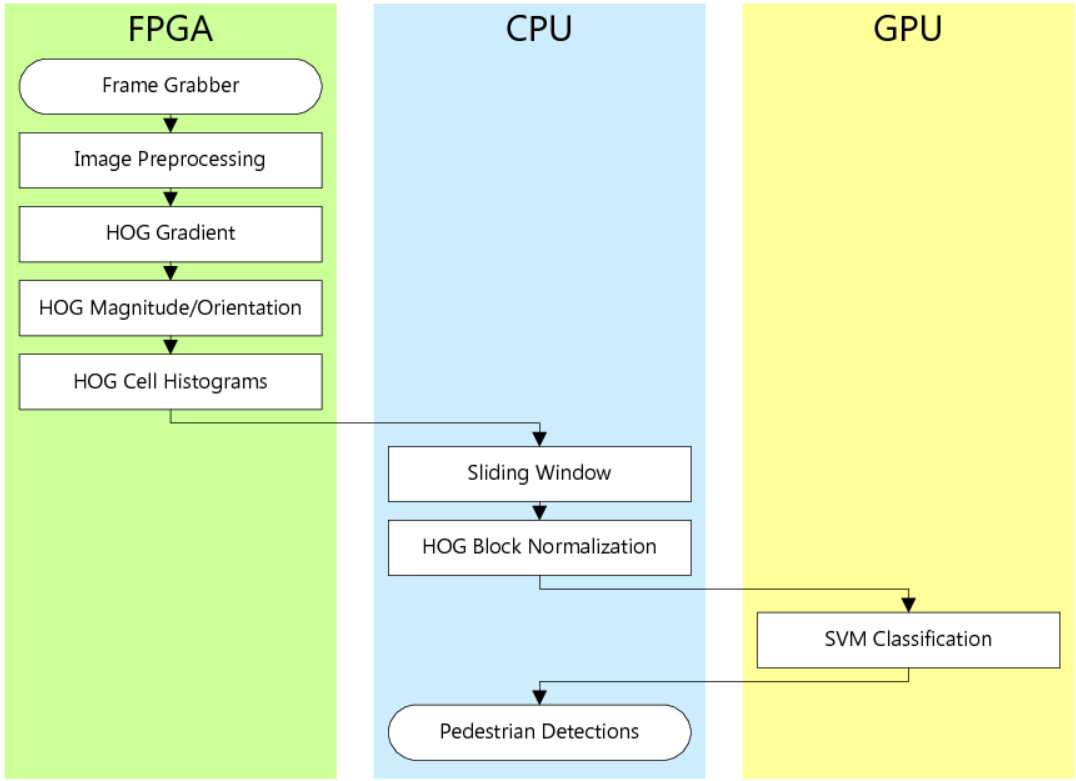
- Classification



- **Machine Learning**
  - Learning from Examples (I/O pairs)
  - Established Pedestrian Datasets [INRIA]
  - **Support Vector Machines (SVM)**
    - Linear Classifier
    - Decision function:
      y = f(**x**) = sgn(**w**·**x**)
    - **w** learned by training



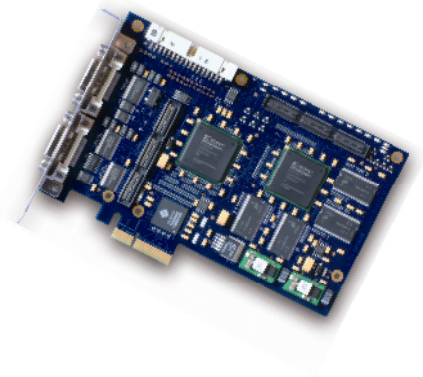
FPGA-CPU-GPU Framework

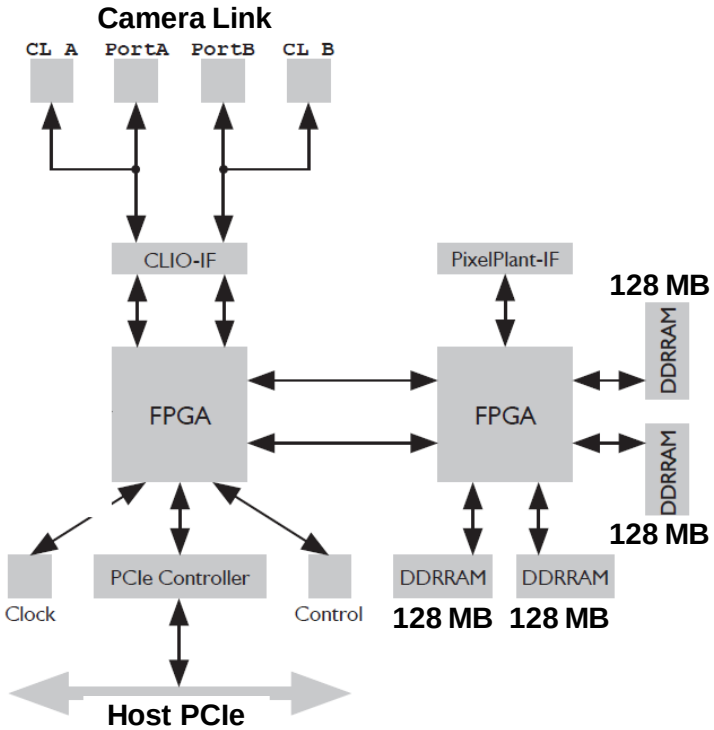


- **FPGA**
  Xilinx Spartan 3 XC3S 4000
- **CPU**
  Intel Core i7, 2.66 GHz
- **GPU**
  NVIDIA GeForce GTX 295

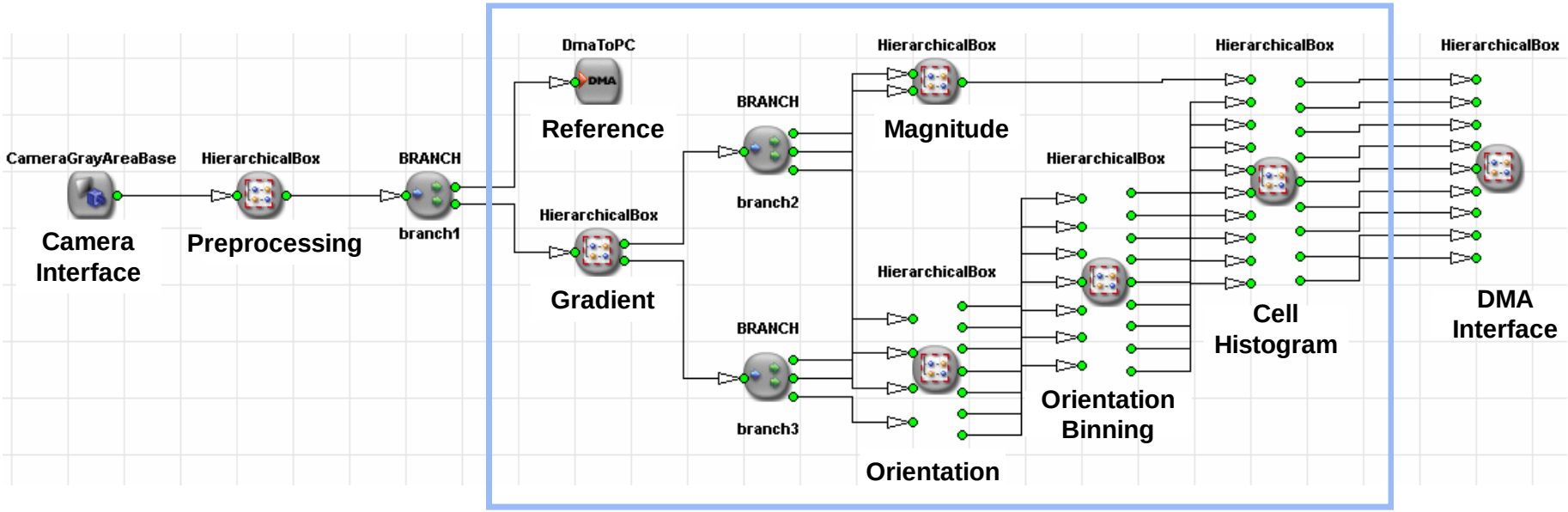
FPGA Rapid Prototyping Platform

- **microEnable IV-FULLx4** [SiliconSoftware]
  - Commercially available frame grabber board
  - FPGA for individual image pre-processing





- FPGA-HOG
  - **VisualApplets**
    - Graphic-oriented hardware development software, based on Xilinx ISE tools
    - Design arranged by image processing operators and transport links
  - **HOG flowchart**



FPGA-HOG

- **Magnitude**

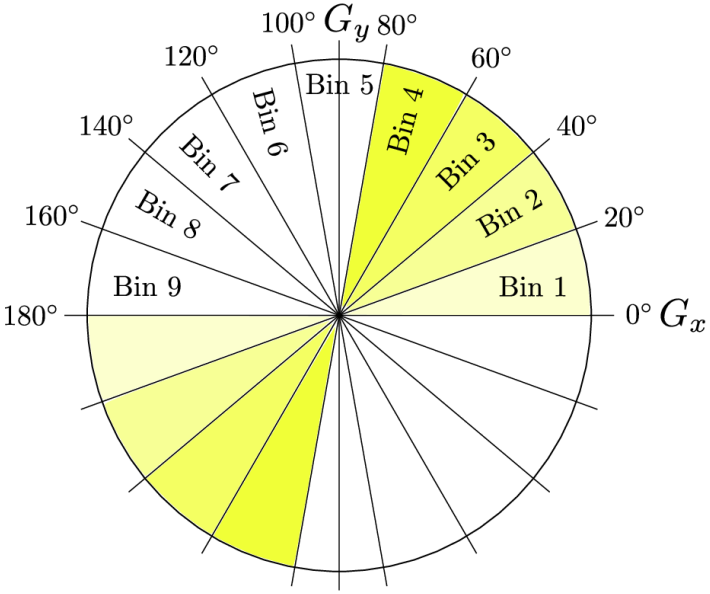
$$|G(x,y)| = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$$

square root extracted,
as Euclidean metric yields best performance

- **Orientation Binning**
  - Example: Bin 1 (0°-20°)

$$Cond.I: \quad G_x(x,y) > 0 \wedge G_y(x,y) > 0 \Leftrightarrow quadrant\ I$$
$$\vee \quad G_x(x,y) < 0 \wedge G_y(x,y) < 0 \Leftrightarrow quadrant\ III$$

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



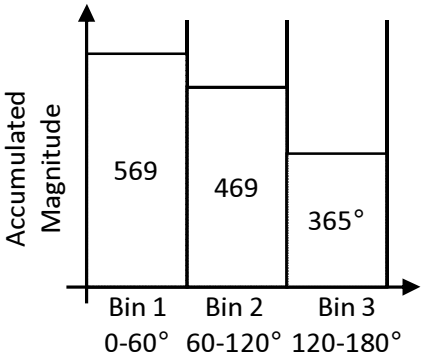
- FPGA-HOG
  - **Histogram Generation**

ϕ

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

Example:

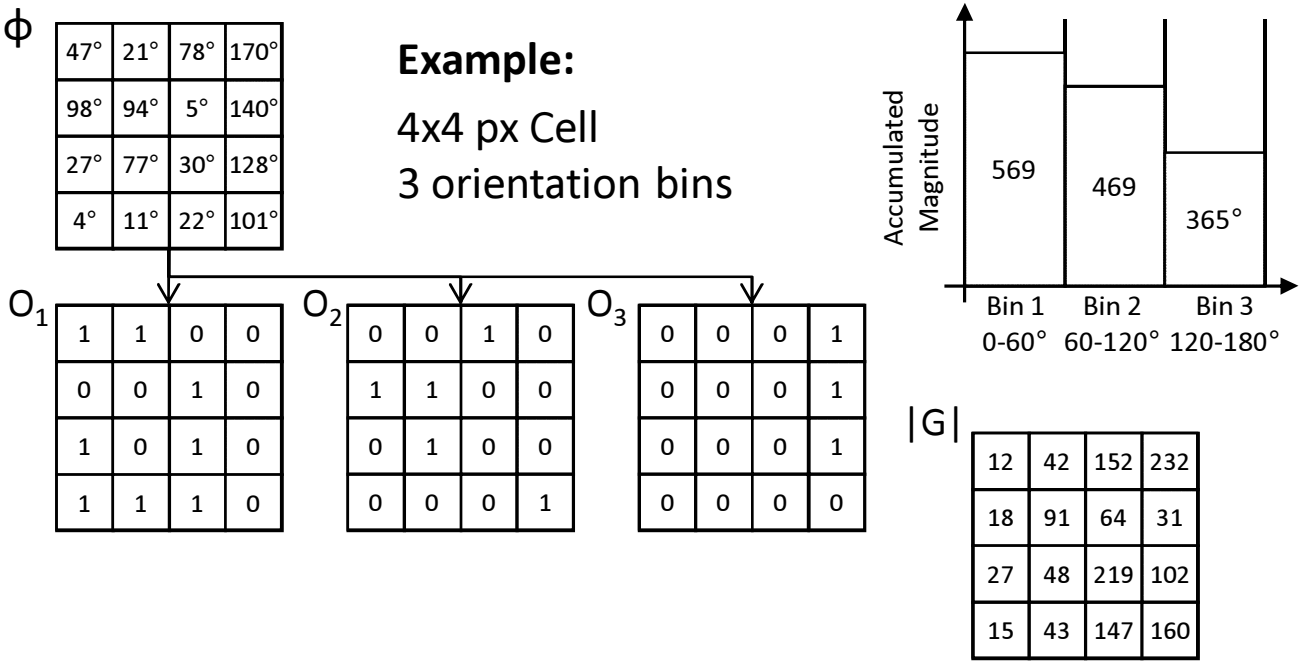
4x4 px Cell
3 orientation bins



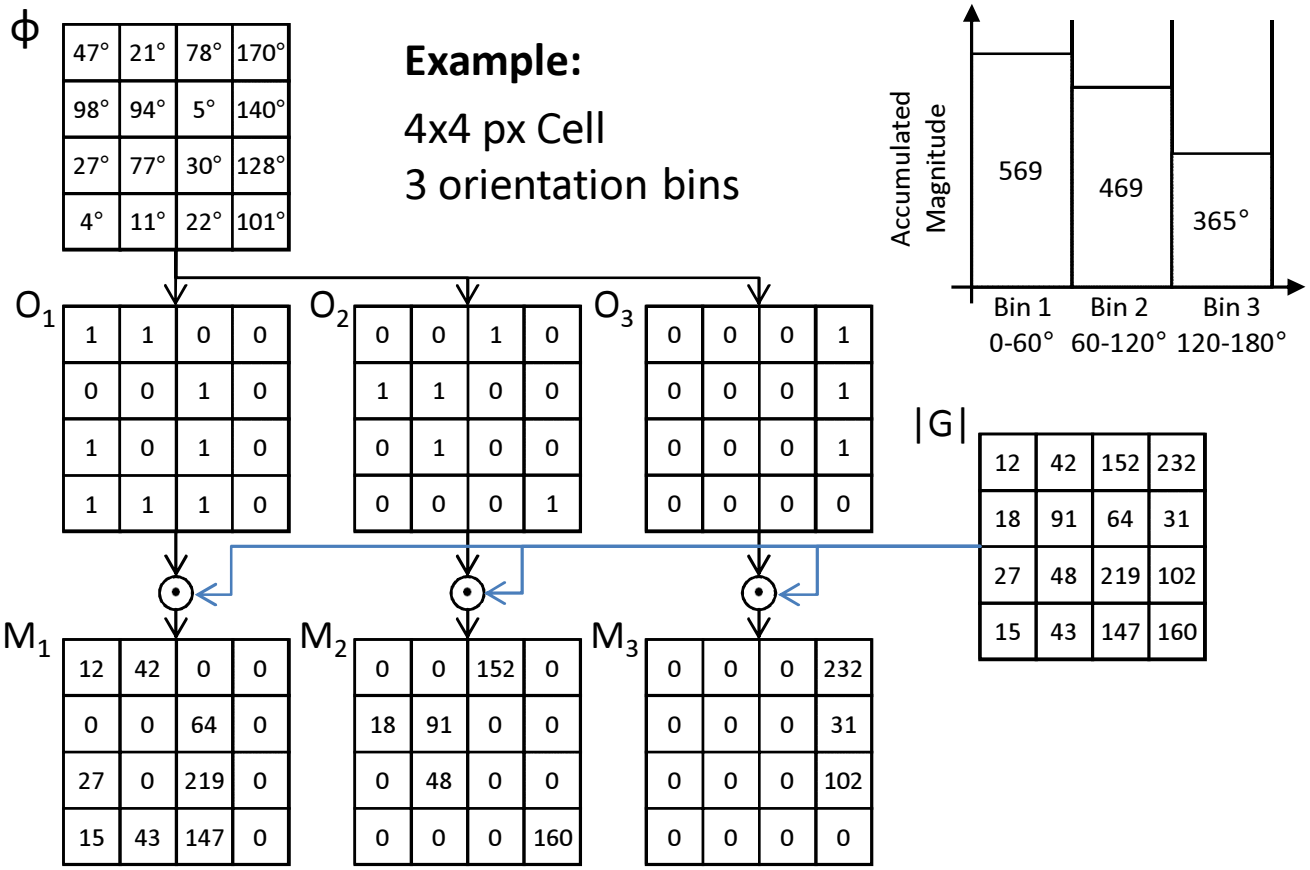
|G|

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

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

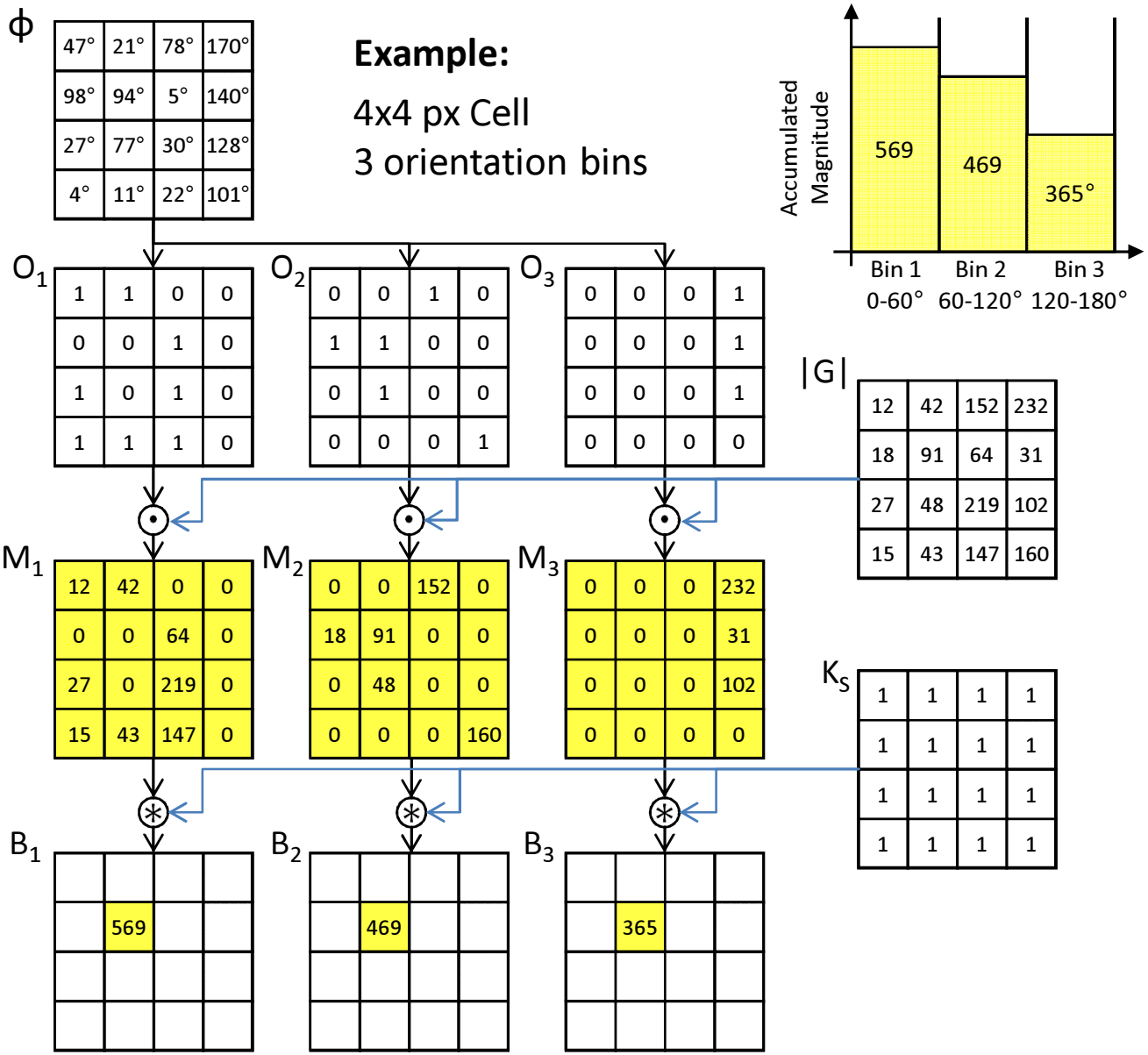


- FPGA-HOG
  - **Histogram Generation**
    - Orientation binning
      - O1: 0-60°
      - O2: 60-120°
      - O3: 120-180°
    - Magnitude weighting



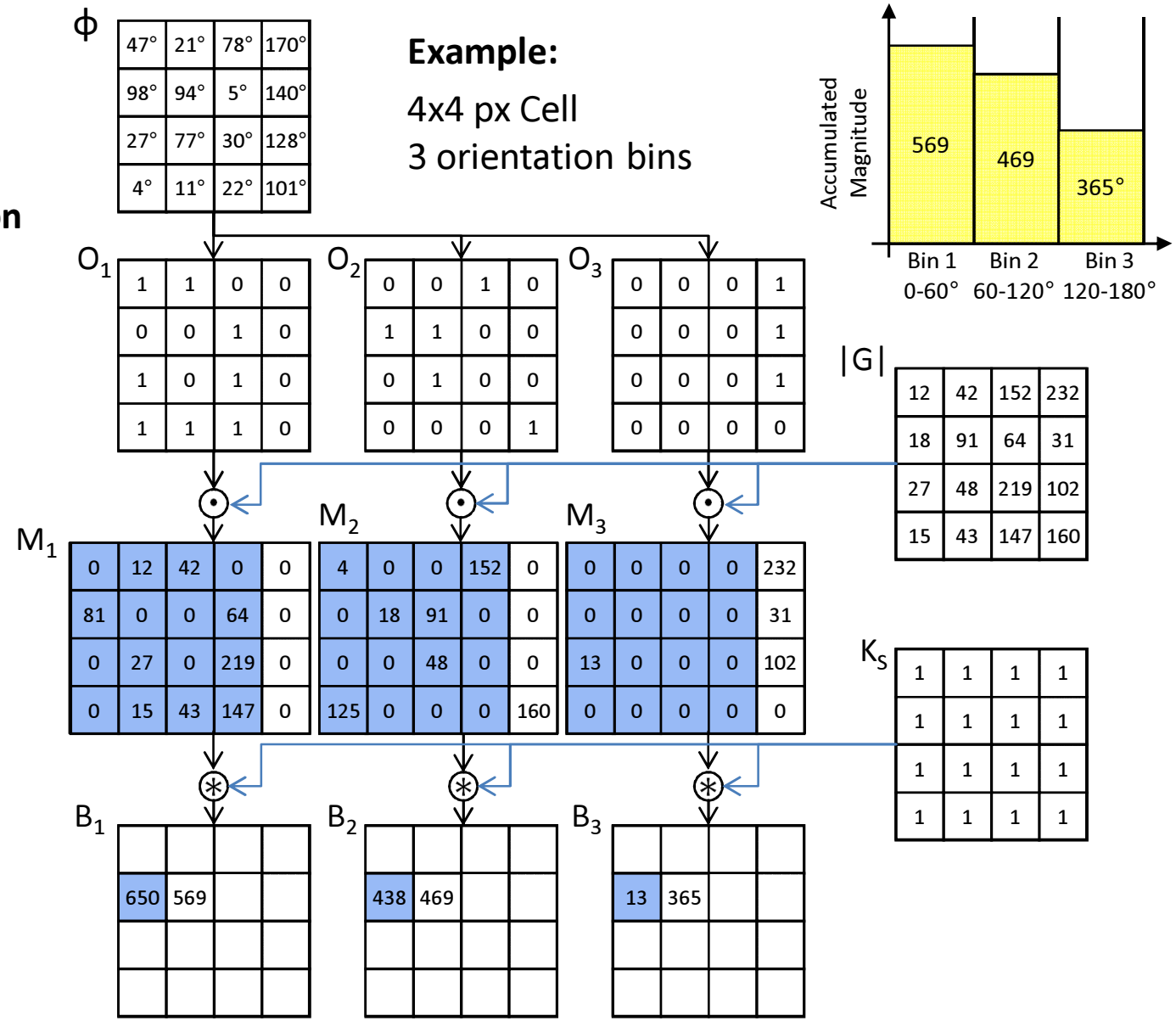
Implementation

- FPGA-HOG
  - **Histogram Generation**
    - Orientation binning
      - O1: 0-60°
      - O2: 60-120°
      - O3: 120-180°
    - Magnitude weighting
    - Histogram bin accumulation



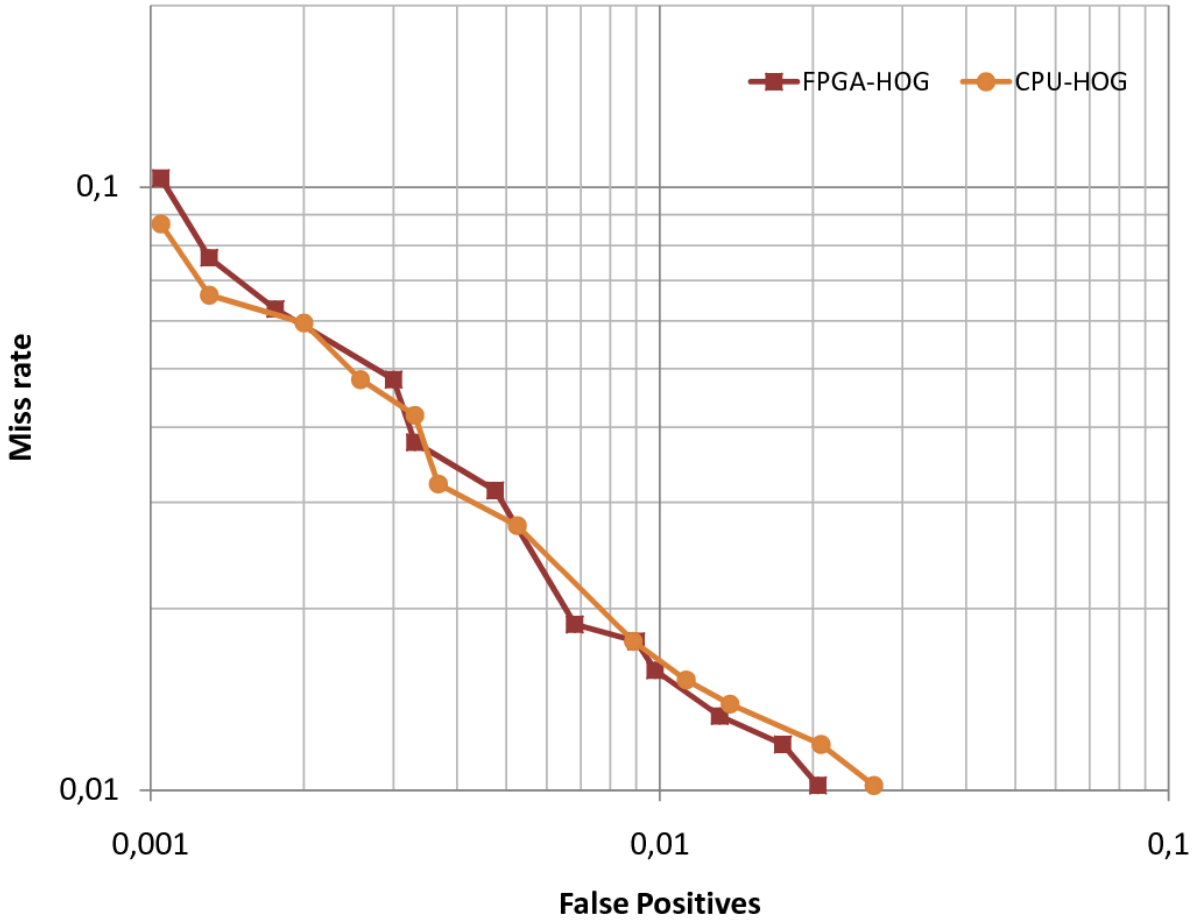
Implementation

- FPGA-HOG
  - **Histogram Generation**
    - Orientation binning
      - O1: 0-60°
      - O2: 60-120°
      - O3: 120-180°
    - Magnitude weighting
    - Histogram bin accumulation



Evaluation

- Classification Performance
  - **Evaluation on [INRIA] benchmark dataset**
  - Detection Error Tradeoff curves (DET)



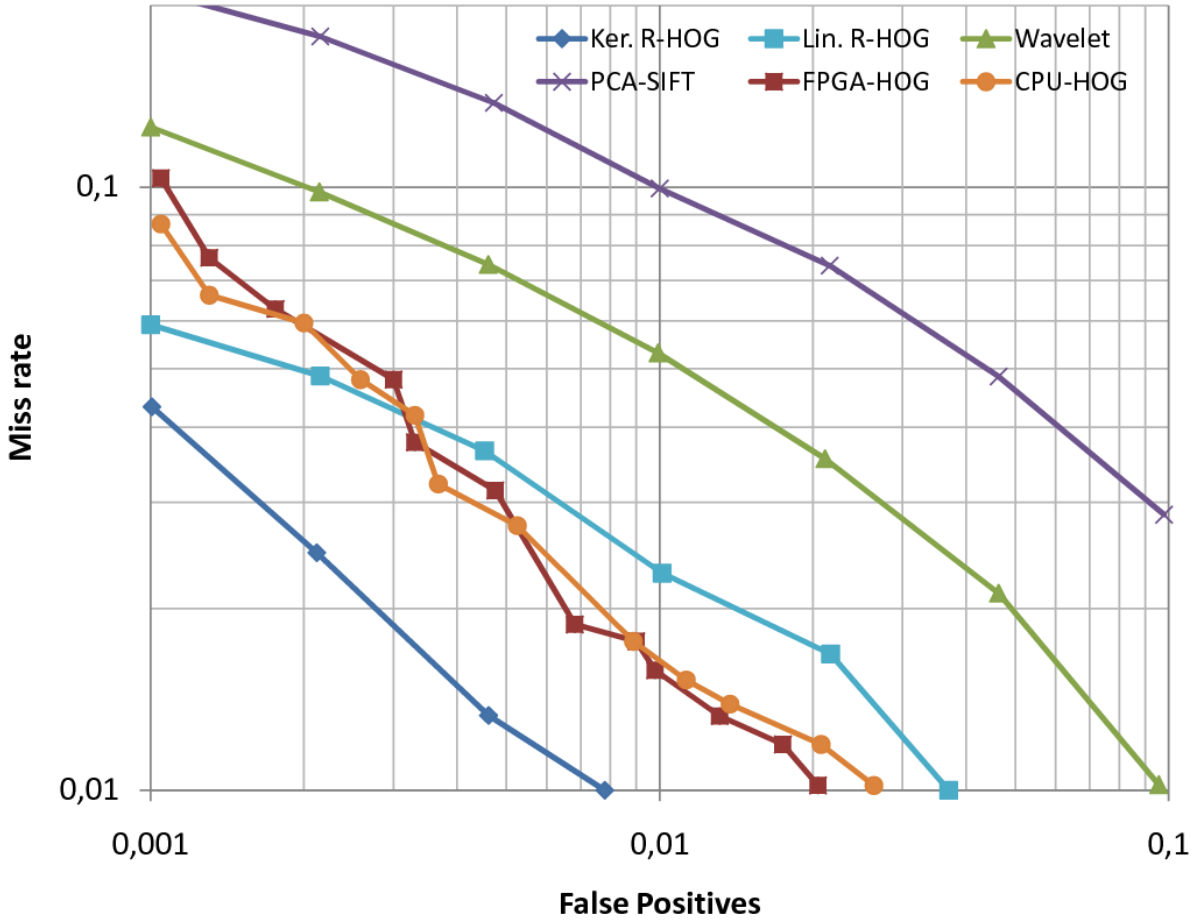
Evaluation

Classification Performance

- **Evaluation on [INRIA] benchmark dataset**
- Detection Error Tradeoff curves (DET)
- **Comparison to other detectors**

  close to reference (Ker. R-HOG)

  without retraining



Evaluation

- Time/Throughput
  - **Latency**

HOG step	Latency [µs] 800x600 px
Image Buffer	26.2
Downsampling	26.0
Gradient	52.0
Magnitude/Orientation	0.1
Histogram	207.2
TOTAL	**311.6 µs**

Evaluation

- Time/Throughput
  - **Latency**

HOG step	Latency [µs] 800x600 px
Image Buffer	26.2
Downsampling	26.0
Gradient	52.0
Magnitude/Orientation	0.1
Histogram	207.2
TOTAL	**311.6 µs**

GPU-HOG [Wojek 2008]
13 ms (320x240 px)

Evaluation

- Time/Throughput
  - **Latency**

HOG step	Latency [µs] 800x600 px
Image Buffer	26.2
Downsampling	26.0
Gradient	52.0
Magnitude/Orientation	0.1
Histogram	207.2
TOTAL	**311.6 µs**

GPU-HOG [Wojek 2008]
13 ms (320x240 px)

  - **Throughput**: full 30 fps (1600x1200 px) delivered by camera

Evaluation

- Time/Throughput
  - **Latency**

HOG step	Latency [µs] 800x600 px
Image Buffer	26.2
Downsampling	26.0
Gradient	52.0
Magnitude/Orientation	0.1
Histogram	207.2
TOTAL	**311.6 µs**

GPU-HOG [Wojek 2008]
13 ms (320x240 px)

  - **Throughput**: full 30 fps (1600x1200 px) delivered by camera

- Resources

Resource Type	Total Number	Usage Level
4 input LUTs	42,435	76%
Internal RAM	60	43%
Embedded Multipliers	18	18%

Xilinx Spartan 3 XC3S 4000
Processing resolution: 800x600

Summary

- What about our Goals?
  - HOG descriptor in real-time on low cost FPGA
  - Good classification results without retraining
  - Real-time pedestrian recognition system for crossroad assistance (20 fps)

Summary

- What about our Goals?
  - HOG descriptor in real-time on low cost FPGA
  - Good classification results without retraining
  - Real-time pedestrian recognition system for crossroad assistance (20 fps)

- Outlook
  - Evaluation with retrained classifier
  - Outsource further parts to FPGA (normalization, SVM)
  - Application for recognition of other traffic participants

Thank you for your attention.

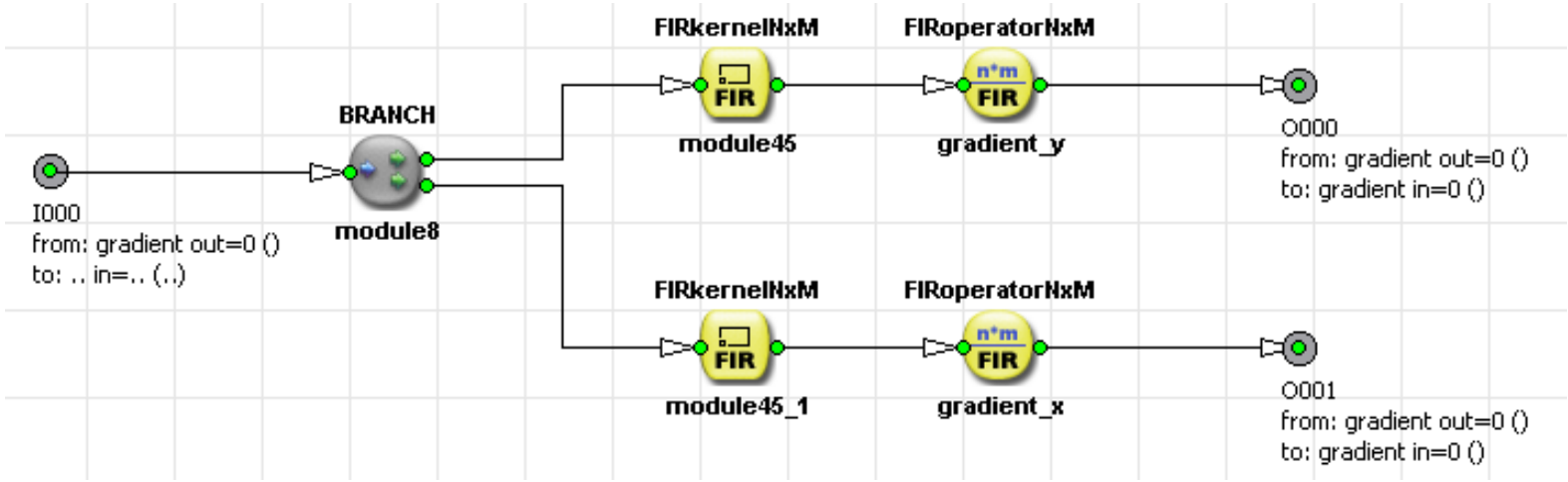
If you have any questions, feel free to ask.

References

- [Dalal::2005] *Histograms of Oriented Gradients for Human Detection*. Dalal, Triggs. CVPR 2005.
- [Cao::2008] *Real-Time Vision-based Stop Sign Detection System on FPGA*. Cao, Deng. DICTA 2008
- [Zhu::2006] *Fast Human Detection Using a Cascade of Histograms of Oriented Gradients*. Zhu, Avidan, Yeh, Cheng. CVPR 2006. MERL.

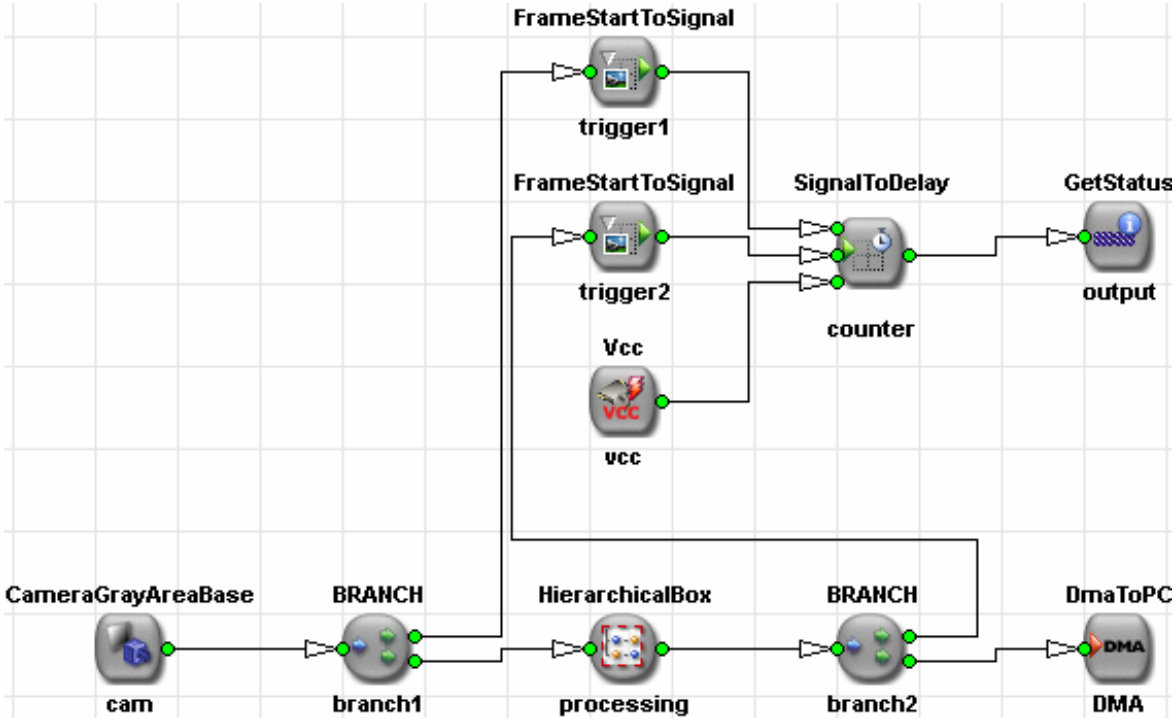
Additional Slides

- VA: Gradient Convolution



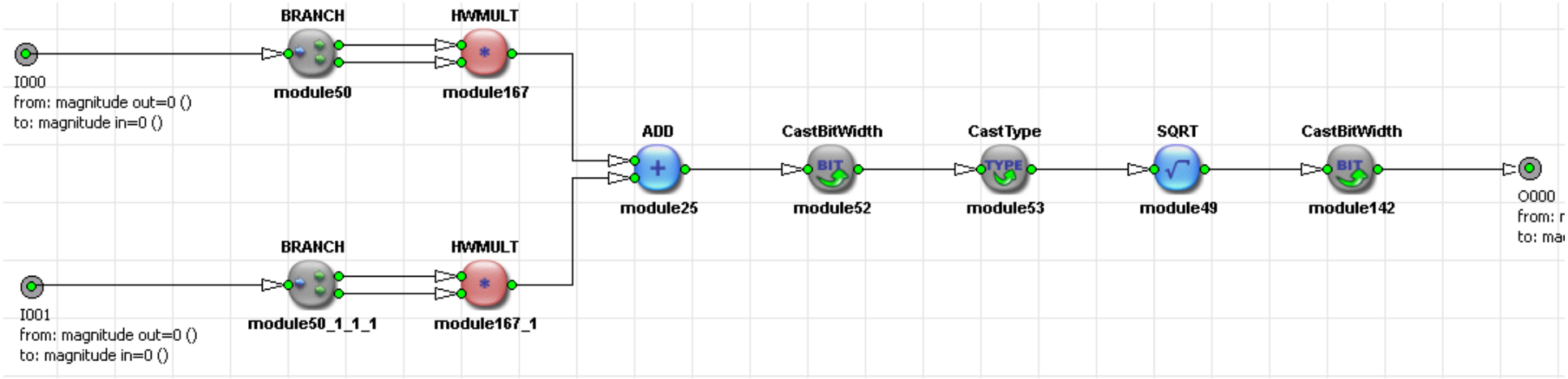
Additional Slides

- VA: Latency Measurement



Additional Slides

- VA: Gradient Magnitude

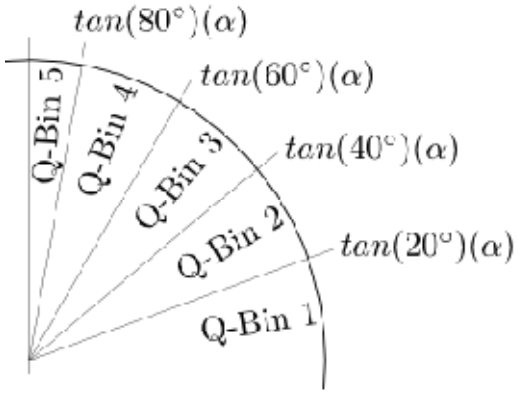


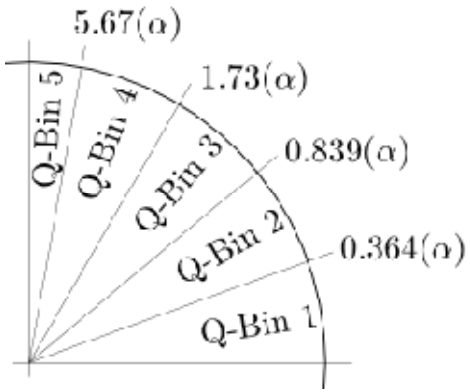
Additional Slides

- How to bypass arctan()

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

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



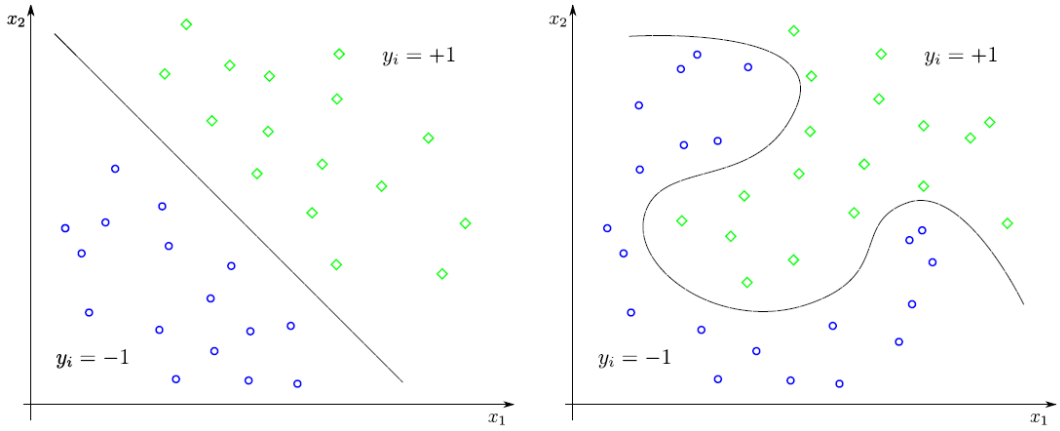


Additional Slides

- HOG Modifications
  - Grayscale images: 1.5% lower detection rate at 1E-4 FPPW (compared to color images)
  - Gradient magnitude: decimal places are lost when the square root is extracted
  - No Anti-Aliasing: Interpolatation of votes between neighbouring histogram bin centres in both orientation and position
  - No Gaussian down-weighting of pixels near the edges of the block: 1% lower detection rate at 1E-4 FPPW

Additional Slides

- Solving Nonlinear Problems with SVMs



  - **Kernel Trick**
    - Decision Function:

$$f(x) = sgn(\sum_{i=1}^{n_S} \alpha_i y_i K(\boldsymbol{s_i}, \boldsymbol{x}) + b_0)$$

Linear Kernels	$\langle \boldsymbol{x} \cdot \boldsymbol{y} \rangle$
Polynomial Kernels	$(\gamma \cdot \langle \boldsymbol{x} \cdot \boldsymbol{y} \rangle + \sigma)^p$
Radial Basis Function (RBF)/Gaussian Kernels	$e^{(-\gamma \|\boldsymbol{x}-\boldsymbol{y}\|^2)}$
Sigmoid Kernels	$tanh(\gamma \cdot \langle \boldsymbol{x} \cdot \boldsymbol{y} \rangle + \sigma)$