INSTITUTE
OF COMMUNICATION,
INFORMATION
AND PERCEPTION
TECHNOLOGIES

Scuola Superiore
Sant'Anna

Smart Camera Networks: advanced applications for next generation Sensor Networks

PhD Defence

Candidate: Claudio Salvadori

Tutor:
Prof. Marco Di Natale

Supervisor:
Dr. Paolo Pagano

October the 29^{th}, 2013

TeCIP Institute, Scuola Superiore Sant'Anna
Area della Ricerca CNR, Via Moruzzi, 1
56127 Pisa, ITALY

Retis
Real-Time Systems Laboratory

Outline

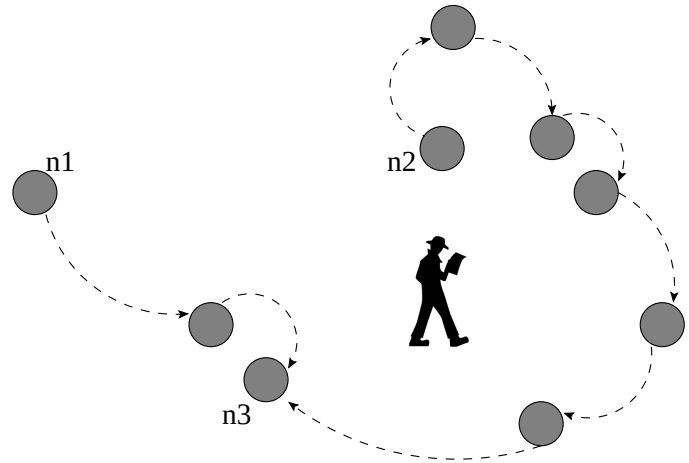
1. Smart Camera Networks
2. Contribution of the presented work
3. Background modeling in embedded systems

Outline

What are Smart Camera Networks?

Smart Camera Networks (*SCN*s) represent the natural evolution of centralized computer vision applications towards full distributed systems.

- In SCNs the application logic is not centralized, but spread among network nodes
- SCN nodes are usually based on low-complexity, low-power and low-cost devices
- Every SCN node has the capability to:
  1. Pre-process images to extract significant features
  2. Aggregate data to understand the surrounding environment



Motivations

Why Smart Cameras?

- On-board processing
- Run-time reconfigurability

Why Smart Camera Networks?

- Large-scale deployments (thus coverage of large areas)
- Solve occlusion problems
- Fault-tolerance w.r.t. single-node failure
- Sensors diversity could be exploited

A Multi-Disciplinary Problem

SCNs involve several disciplines...

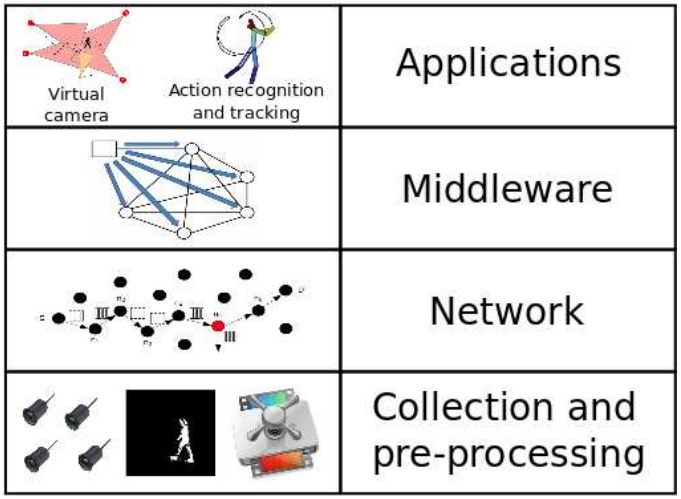
- Networking
- Signal Processing
- Computer Vision
- Embedded Systems Design

Outline

SCNs in future Internet

We propose the vision of a SCN based on the *Internet of Things* (or *IoT*) paradigm [PSM+12]:

- Every single computational element installed inside the SC can be abstracted as resource and made available to other network nodes
- The system can instantiate any computer vision pipeline by connecting these functional blocks (model-based design reasoning)
- Every distributed visual application is designed "*as a service*" by the use of a *Middleware of Things* (*MoT*)



Smart Cameras requirements

In this environment, a Smart Camera has to:

- Satisfy the flexibility requirements of the MoT
- Handle heavy computer vision processing tasks (e.g., pixel-wise operations, machine learning algorithms)

Smart Cameras requirements

In this environment, a Smart Camera has to:

- Satisfy the flexibility requirements of the MoT
- Handle heavy computer vision processing tasks (e.g., pixel-wise operations, machine learning algorithms)

Thus, we imagine a SCN as a heterogeneous group of Smart Cameras:

- Connected using different inter-operable communication protocols (e.g., IEEE802.15.4, WiFi, VLC, Ethernet)
- Based on several types of chips, to perform operations at different levels of complexity (e.g., micro-controllers, FPGAs, ...)

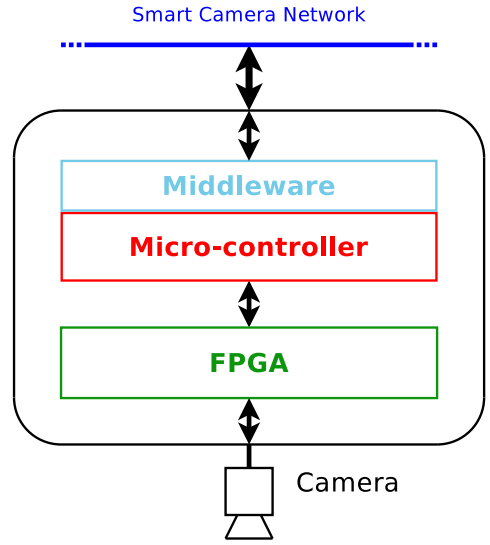
Smart Cameras

All the works proposed in this thesis are addressed to exploit totally the potentiality of micro-controllers (e.g., PIC32), but we have demonstrated that these chips can not handle the deployment of complex state-of-the art algorithms (e.g., HOG, SVM, Neural Networks)...

Smart Cameras

All the works proposed in this thesis are addressed to exploit totally the potentiality of micro-controllers (e.g., PIC32), but we have demonstrated that these chips can not handle the deployment of complex state-of-the art algorithms (e.g., HOG, SVM, Neural Networks)...
... For these reasons, we have design the architecture of the future SC [MSP+13], which is based on:

- A *microcontroller*: to handle both the high level operations and the network communications
- A *FPGA*: to process complex computer vision algorithms (i.e., pixel-wise operations and/or machine learning blocks)



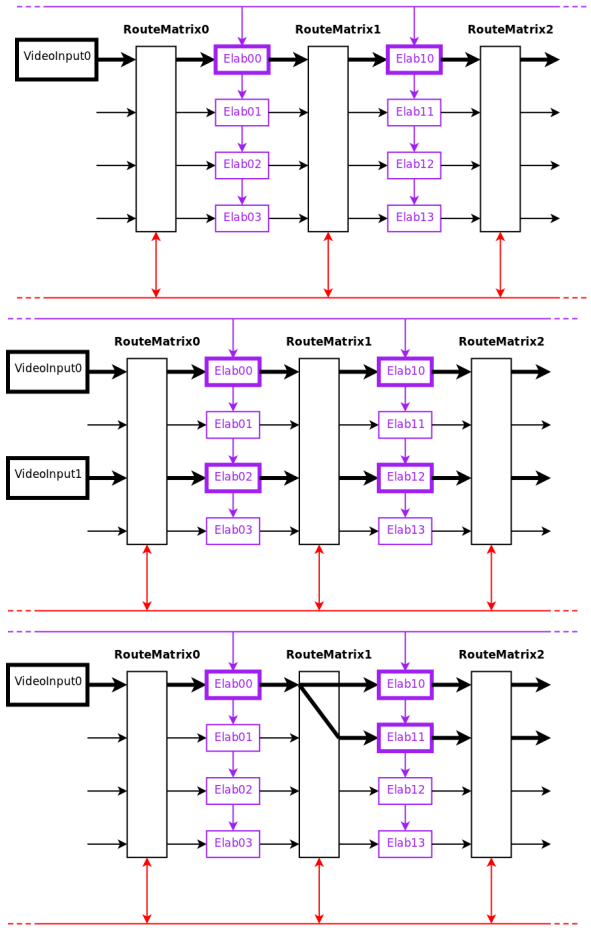
FPGA internal architecture

FPGAs are usually exploited in the realisation of dedicated solutions to optimise the low level operations, neglecting both the reconfigurability and flexibility issues.

The proposed FPGA internal architecture can:

- Define at run-time the routing path of a data stream
- Handle the execution of parallel pipelines with different data sources (multi-camera)
- Split a data stream in two or more pipelines

The IoT requirements are satisfied!



Contribution of the presented work

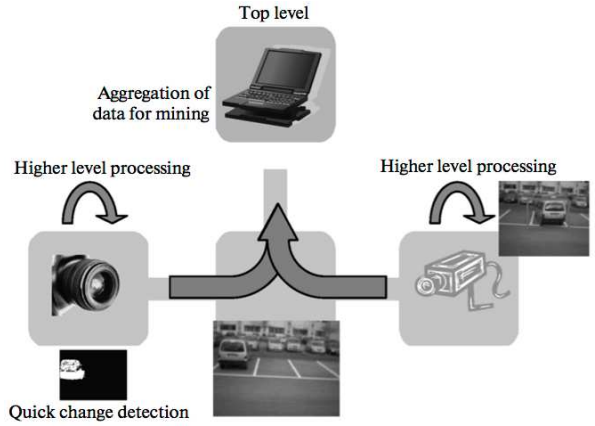
On-board processing for micro-controllers based Smart Cameras

Parking spaces monitoring

Solution 1:

A multi-tier approach [MMN+11, MMP+10]:

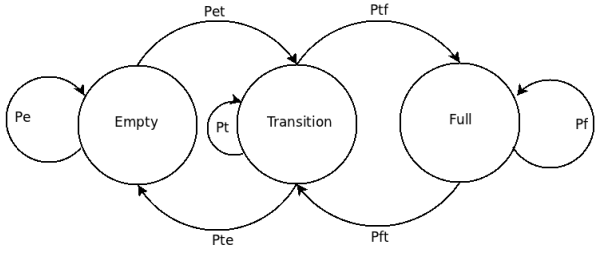
- Fast detection algorithm
- Cascade of binary classifiers based on vector of Haar-like features



Solution 2:

Background modeling based on a three states Markov chain [SPGP12]:

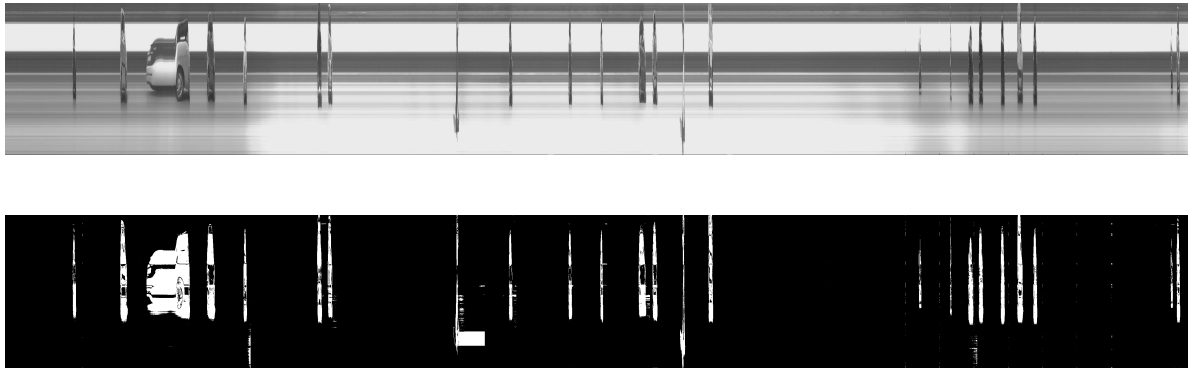
- Objects absorption synchronous with the chain
- Running averages for light changes adaptation



The vehicles counter application

In this application the lack of resources problem is addressed by processing a smaller amount of data, while taking benefit from the environmental conditions given by the vehicles movements [CSP+10, CSP+12].

- Based on the *Linesensor theory*:
  - Linesensor is a camera-based sensor capable to acquire and process 1D vectors of pixels
- Background subtraction paradigm:
  - It needs an adaptive background model
  - Single Metric Model or *SMM*: the whole background is modeled considering the statistical distribution of a metric

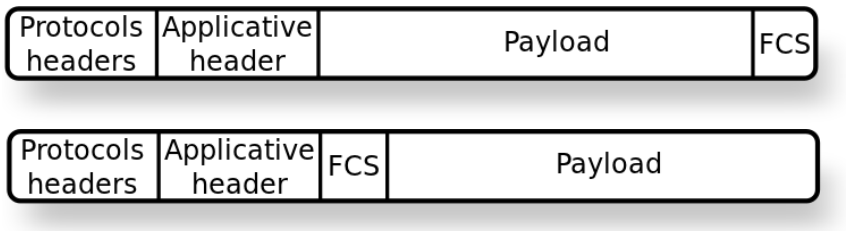


Video surveillance advanced applications

How can we send images through IEEE802.15.4 network using low-cost and low complexity Smart Cameras? Which is the impact on the image quality?

Steaming raw images [PGS+11]:

- Innate error resiliency of raw pixels (no correlation on their representation)
  - New MAC data message

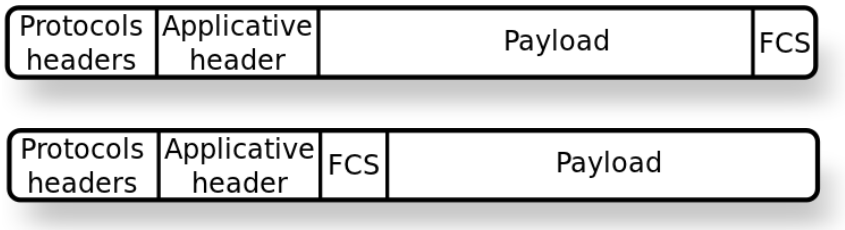


Video surveillance advanced applications

How can we send images through IEEE802.15.4 network using low-cost and low complexity Smart Cameras? Which is the impact on the image quality?

Steaming raw images [PGS+11]:

- Innate error resiliency of raw pixels (no correlation on their representation)
  - New MAC data message

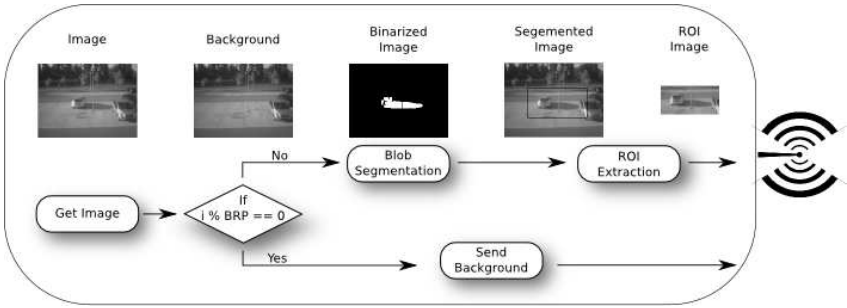


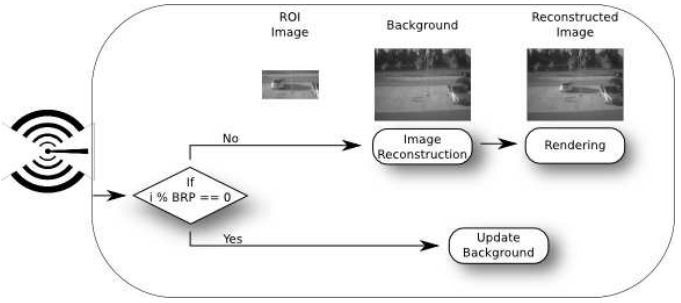
- Drawback: Using entire raw images only one stream can instantiated
  - Data compression

Video surveillance advanced applications

We proposed a low-complexity codec [SPP+12, SPM+13]:

- Transmitter side:
  - Background subtraction based
  - Considers only the zones of the images where a change is detected
  - Every N frames a new reference image is sent $\Rightarrow$ the codec robustness on light changes is enhanced
- Receiver side:
  - Image reconstruction
  - Three light-weight concealments (black, background and copy)





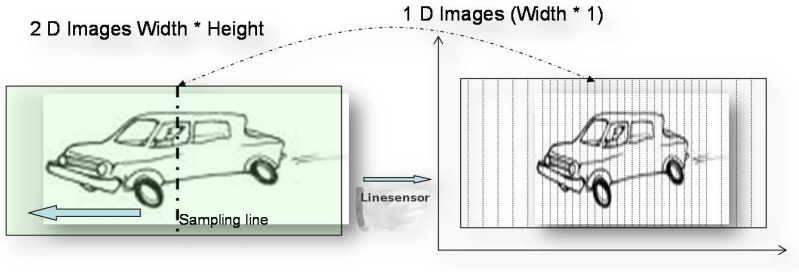
Computer vision algorithms on low-complexity embedded systems

Our target is to port **advanced** computer vision algorithms on embedded systems:

- Low-computation capabilities (e.g., PIC32 $\Rightarrow$ 80Mhz)
- Low-memory (e.g., PIC32 $\Rightarrow$ 128KB)
- No Floating Point Unit (FPU)

Typical approach:

- Low complexity algorithms
- Low resolution images or part of them (i.e., RoIs)



Gaussian Mixture Model on low-complexity embedded systems

We propose to optimise and/or approximate the state-of-the-art computer vision algorithms in order to:

- Reduce the memory footprint:
  - compressing the representation of the commonly used data types
- speed up the data processing:
  - Using of integer representation

Gaussian Mixture Model on low-complexity embedded systems

We propose to optimise and/or approximate the state-of-the-art computer vision algorithms in order to:

- Reduce the memory footprint:
  - compressing the representation of the commonly used data types
- speed up the data processing:
  - Using of integer representation

We propose two novel integer computer arithmetic techniques to update Gaussian parameters for Gaussian mixture model (GMM) algorithm [SMP+12]:

- The mean value and the variance of each Gaussian are updated by a redefined and generalised "round" operation
- Weights are represented by counters and updated following stochastic rules

Outline

Background/foreground segmentation

Background/foreground segmentation:

- The capability of extracting moving objects from a video sequence captured using a static camera
  - For example background subtraction

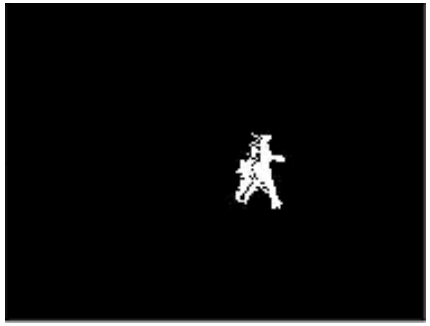
$$\mathbf{O} = |\mathbf{B} - \mathbf{I}|_{TH} \tag{1}$$

  - Necessity of a background model to be robust to light changes and stable geometrical changes
- First step in video-surveillance pipelines

Background subtraction in embedded devices

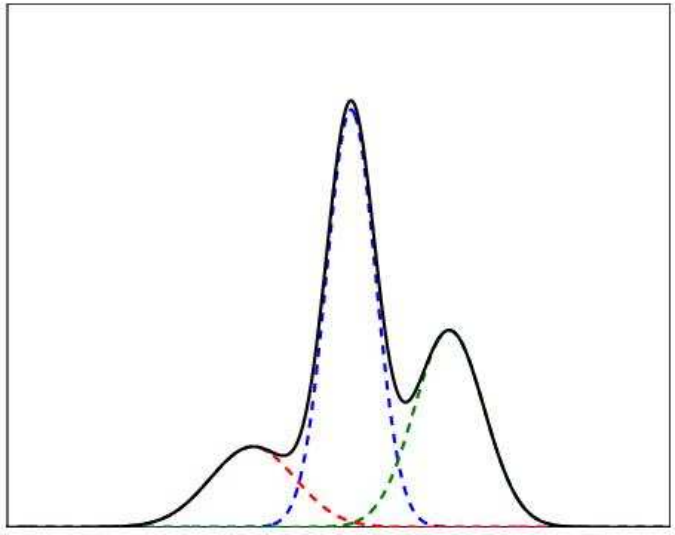
In embedded devices (i.e., Smart Cameras):

- Low-complexity and low-memory footprint algorithms are usually considered
  - Median filter
  - Running averages
- Only one parameter is used to model the background
  - The foreground is not explicitly modeled
  - Possible formation of artifacts into the binarized images
  - **Solution**: usage of multi-modal techniques



Gaussian Mixture Model

- Each background pixel is modeled using a Mixture of G Gaussians



- Each Gaussian g is represented by 3 parameters: the mean value ($\mu_{i,g}(j)$), the variance ($\sigma^2_{i,g}(j)$) and the weight ($w_{i,g}(j)$)

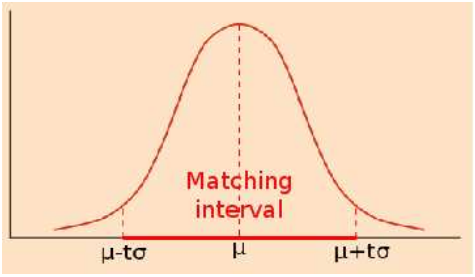
μ	σ^2	w
[0, 255]	$[\sigma^2_{min}, \sigma^2_{MAX}]$	[0, 1]

Gaussian Mixture Model

- A pixel $p_i(j)$ matches the g-th Gaussian if:

$$d^2_{i,g}(j) < T\sigma^2_{i,g}(j) \qquad (2)$$

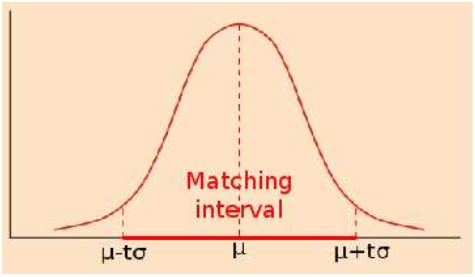
- We call this relation *membership criterion*



Gaussian Mixture Model

- A pixel $p_i(j)$ matches the g-th Gaussian if:

$$d^2_{i,g}(j) < T\sigma^2_{i,g}(j) \qquad (2)$$

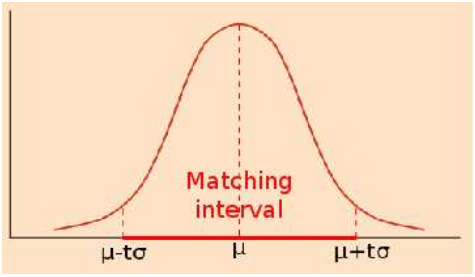


- We call this relation *membership criterion*
- Background is modeled by the most popular Gaussians, while foreground is modeled by the rest

Gaussian Mixture Model

- A pixel $p_i(j)$ matches the g-th Gaussian if:

$$d^2_{i,g}(j) < T\sigma^2_{i,g}(j) \qquad (2)$$



- We call this relation *membership criterion*
- Background is modeled by the most popular Gaussians, while foreground is modeled by the rest
- The model has to be adaptive to the light and stable geometrical changes

Gaussian Mixture Model

Given α (learning rate), the Gaussians parameters are updated using FIRs:

- If the pixel $p_i(j)$ matches the g-th Gaussian:

$$\mu_{i,g}(j+1) = \mu_{i,g}(j) + k_{i,g}(j)^1 d_{i,g}(j) \quad (3)$$

$$\sigma^2_{i,g}(j+1) = \sigma^2_{i,g}(j) + k_{i,g}(j)(d^2_{i,g}(j) - \sigma^2_{i,g}(j)) \quad (4)$$

$$w_{i,g}(j+1) = (1-\alpha) \cdot w_{i,g}(j) + \alpha \quad (5)$$

- If the pixel $p_i(j)$ does not match the g-th Gaussian:

$$w_{i,g}(j+1) = (1-\alpha) \cdot w_{i,g}(j) \quad (6)$$

[1] **Note**: Where $k_{i,g}(j) = \dfrac{\alpha}{w_{i,g}(j)}$

GMM vs. Micro-controllers

Is it possible to port GMM on PIC32?

PIC32:

- RAM: 128KB
- Clock frequency: 80MHz

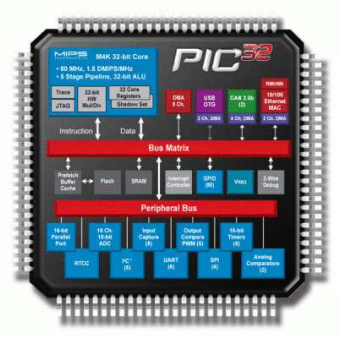


Table: Algorithm footprint (QQ-VGA images)

Precision	Bytes per Gaussian	Footprint 2G (Bytes)	Footprint 3G (Bytes)
Double	24	921,600	1,382,400
Float	12	460,800	691,200
uint8_t	3	115,200	172,800

Gaussian Mixture Model vs. Micro-controllers

Sub-integer representation of the Gaussian parameters:

- We represent each parameter with less than 8bits
- A method to map floating-point number has to be found

Gaussian Mixture Model vs. Micro-controllers

Sub-integer representation of the Gaussian parameters:

- We represent each parameter with less than 8bits
- A method to map floating-point number has to be found

Thus:

- The mean value and the variance updating that depend directly on the pixel values, will be handled defining a *generalised round operation*
- The weights are represented as counters and updated using a stochastic technique

Mean value and variance updating

The generalised-round operator is thought to enable a satisfactorily accurate representation of floating-point numbers in the sub-integer domain $\mathbb{U}$, with $\mathbb{U} \subseteq \mathbb{Z}$

Mean value and variance updating

The generalised-round operator is thought to enable a satisfactorily accurate representation of floating-point numbers in the sub-integer domain $\mathbb{U}$, with $\mathbb{U} \subseteq \mathbb{Z}$

- The sub-integer domain $\mathbb{U}$:
  - can be described as the *discrete* and *ordered* set of numbers $\mathbf{B} = \{b_m \in \mathbb{U} | b_{m+1} - b_m = \gamma\}$;
  - can be characterized by two parameters: the *granularity* γ, and the *updating step* ξ, defined as a function of γ:

$$\xi = \mathcal{F}(\gamma) = R\gamma \tag{7}$$

  - $R \in (0, 1)$ is called *rounding parameter*

Mean value and variance updating

The generalised-round operator is thought to enable a satisfactorily accurate representation of floating-point numbers in the sub-integer domain $\mathbb{U}$, with $\mathbb{U} \subseteq \mathbb{Z}$

- The sub-integer domain $\mathbb{U}$:
  - can be described as the *discrete* and *ordered* set of numbers $\mathbf{B} = \{b_m \in \mathbb{U} | b_{m+1} - b_m = \gamma\}$;
  - can be characterized by two parameters: the *granularity* γ, and the *updating step* ξ, defined as a function of γ:

$$\xi = \mathcal{F}(\gamma) = R\gamma \tag{7}$$

  - $R \in (0, 1)$ is called *rounding parameter*
- The γ-floor operator $\lfloor \circ \rfloor_\gamma$ is defined as:

$$\lfloor \delta \rfloor_\gamma = \lfloor \frac{\delta}{\gamma} \rfloor \gamma \tag{8}$$

Mean value and variance updating

Example 1:

- Suppose to consider the integer numbers ($\mathbb{U} \equiv \mathbb{Z}$)
- $\gamma = 1$
- Usually $R = 0.5$ is consider in order to discriminate the round-up from the round-down
- So:
  - $2.4 \Rightarrow 2$
  - $2.5 \Rightarrow 3$

Mean value and variance updating

Example 1:

- Suppose to consider the integer numbers ($\mathbb{U} \equiv \mathbb{Z}$)
- $\gamma = 1$
- Usually $R = 0.5$ is consider in order to discriminate the round-up from the round-down
- So:
  - $2.4 \Rightarrow 2$
  - $2.5 \Rightarrow 3$

Example 2:

- Suppose $\gamma = 2$ and $\delta = 9$
- $\lfloor \delta \rceil_\gamma = \lfloor \frac{\delta}{\gamma} \rceil \gamma = \lfloor \frac{9}{2} \rceil 2 = 8$

Calculating the updating step ξ

In Eq. 7 we have stated that the updating step ξ directly depends on both the updating step γ and the rounding parameter R
Considering a constant R:

- Limitations in the learning rate range...
  - The updating contribution depends on $k_{i,g}(j) = \frac{\alpha}{w_{i,g}(j)}$
- Updating the mean value and the variance is possible for only a sub-set of learning rate values

Calculating the updating step ξ

In Eq. 7 we have stated that the updating step ξ directly depends on both the updating step γ and the rounding parameter R
Considering a constant R:

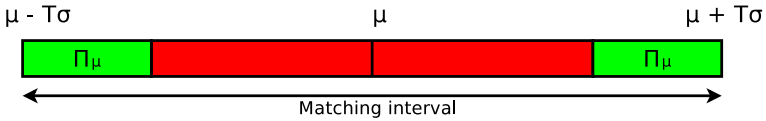
- Limitations in the learning rate range...
  - The updating contribution depends on $k_{i,g}(j) = \frac{\alpha}{w_{i,g}(j)}$
- Updating the mean value and the variance is possible for only a sub-set of learning rate values

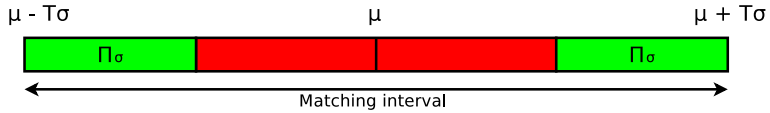
We suggest a more general technique:

- the updating step ξ is computed as a function of the learning rate α, to
  - Ensure appropriate updating of both parameters
  - Maintain the membership intervals of Gaussians similar to the ones estimated by the floating-point approach

Calculating the updating step ξ

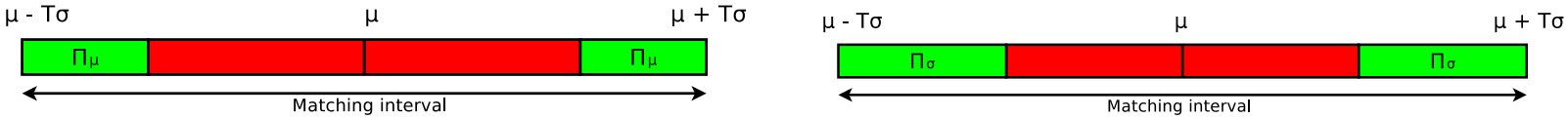
The idea is to define critical intervals close to the border of the matching interval





Calculating the updating step ξ

The idea is to define critical intervals close to the border of the matching interval

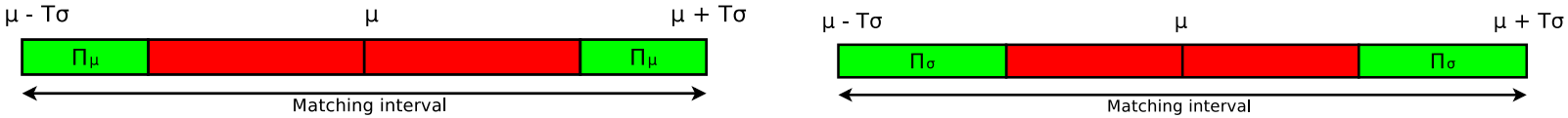


Thus, if the pixel p_i matches the interval...

- ... Π_μ, the mean value is updated
- ... Π_σ, the variance is updated

Calculating the updating step ξ

The idea is to define critical intervals close to the border of the matching interval



Thus, if the pixel p_i matches the interval...

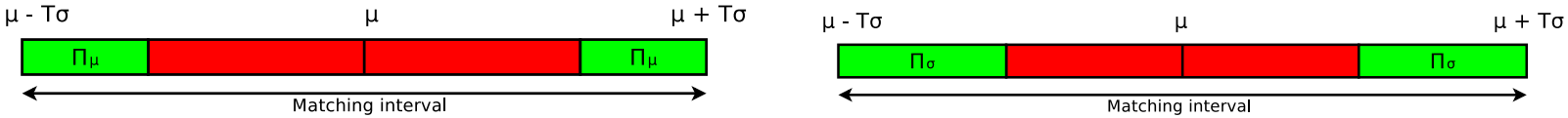
- ... Π_μ, the mean value is updated
- ... Π_σ, the variance is updated

The width of both the critical areas:

- Are calibrated iteratively in order to emulate the GMM matching intervals
- Are functions of the learning rate

Calculating the updating step ξ

The idea is to define critical intervals close to the border of the matching interval



Thus, if the pixel p_i matches the interval...

- ... Π_μ, the mean value is updated
- ... Π_σ, the variance is updated

The width of both the critical areas:

- Are calibrated iteratively in order to emulate the GMM matching intervals
- Are functions of the learning rate

The generalised round operation is defined as:

$$G\_ROUND(\delta, \xi, \gamma, C) = \begin{cases} \lfloor \delta \rceil_\gamma & \text{if } |\delta| \geq \gamma \\ \gamma & \text{if } (|\delta| < \gamma) \wedge (\delta \in \Pi) \\ 0 & \text{if } (|\delta| < \gamma) \wedge (\delta \notin \Pi) \end{cases} \quad (9)$$

Weight updating

The weight updating rule does not depend directly on the value of the pixel p_i, but only on the membership to a certain Gaussian:

- If the pixel matches to a Gaussian, its weight is increased by a given value

$$w_{i,g}(j+1) = (1-\alpha) \cdot w_{i,g}(j) + \alpha \tag{10}$$

- If the pixel does not match to a Gaussian, its weight is decreased by a given value

$$w_{i,g}(j+1) = (1-\alpha) \cdot w_{i,g}(j) \tag{11}$$

Weight updating

The weight updating rule does not depend directly on the value of the pixel p_i, but only on the membership to a certain Gaussian:

- If the pixel matches to a Gaussian, its weight is increased by a given value

$$w_{i,g}(j+1) = (1-\alpha) \cdot w_{i,g}(j) + \alpha \tag{10}$$

- If the pixel does not match to a Gaussian, its weight is decreased by a given value

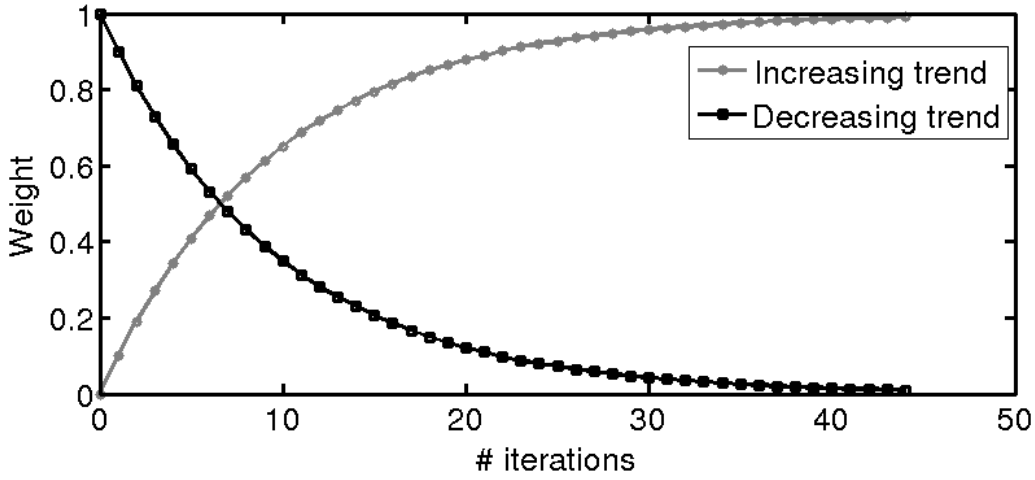
$$w_{i,g}(j+1) = (1-\alpha) \cdot w_{i,g}(j) \tag{11}$$

Solving these equations (starting from 0 and 1 respectively):

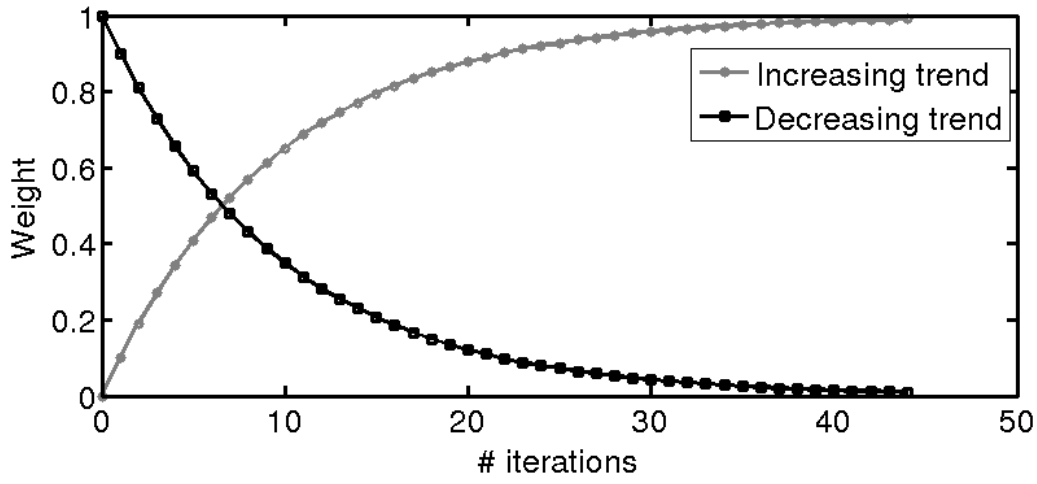
$$w_g(s) = 1 - (1-\alpha)^s \tag{12}$$

$$w_g(s) = (1-\alpha)^s \tag{13}$$

Weight updating



Weight updating



Solving Eq. 12 (the increasing trend):

$$s(w, \alpha) = \frac{\log_{10}(1 - w)}{\log_{10}(1 - \alpha)} \tag{14}$$

It is possible to represent a weight as an integer counter and represent it using a fixed number of l bits, able to cover the range of values $[0, L]$ (where $L = 2^l - 1$)

How to compute the weights from the counters?

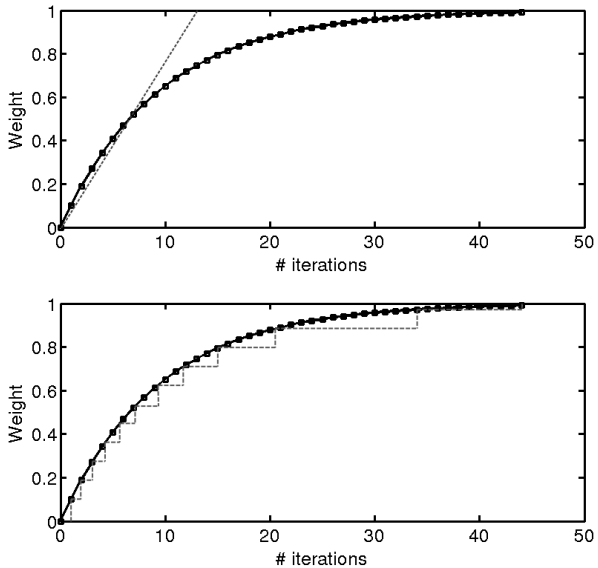
- Though these considerations simplify the weight updating problem...
- ... the operation to retrieve the weight value continues to be floating point based because of the exponential nature of the considered equations

How to compute the weights from the counters?

- Though these considerations simplify the weight updating problem...
- ... the operation to retrieve the weight value continues to be floating point based because of the exponential nature of the considered equations

Two options are considered to simplify/approximate:

- A line (*iGMM-l*)
- A staircase (*iGMM-s*)



Stochastic weights updating

- The number of steps to retrieve the highest value of weight (namely $1 - \epsilon$ where ϵ is a very small positive number, e.g. $\epsilon = 0.01$) is inversely proportional to the learning rate
- We consider a limited number of bits l to represent the weights
- The deterministic calculation of both the linear and staircase trends could be efficient only for large values of α
- To overcome such restriction, we propose a stochastic method

Stochastic weights updating

Suppose that:

- The counter have to be updated by the value *STEP*
- The number of iterations to reach the highest value of weight for a fixed value of learning rate α is A
- The available bits to represent the weight are l, such that

$$L = 2^l - 1 < A \tag{15}$$

We propose to update the integer counter s of a *STEP* amount with a certain probability to reach the highest value of weight in A steps in average!

Stochastic weights updating

Suppose that:

- The counter have to be updated by the value $STEP$
- The number of iterations to reach the highest value of weight for a fixed value of learning rate α is A
- The available bits to represent the weight are l, such that

$$L = 2^l - 1 < A \tag{15}$$

We propose to update the integer counter s of a $STEP$ amount with a certain probability to reach the highest value of weight in A steps in average!

$$s_n = s_n + \mathbf{H}_n \tag{16}$$

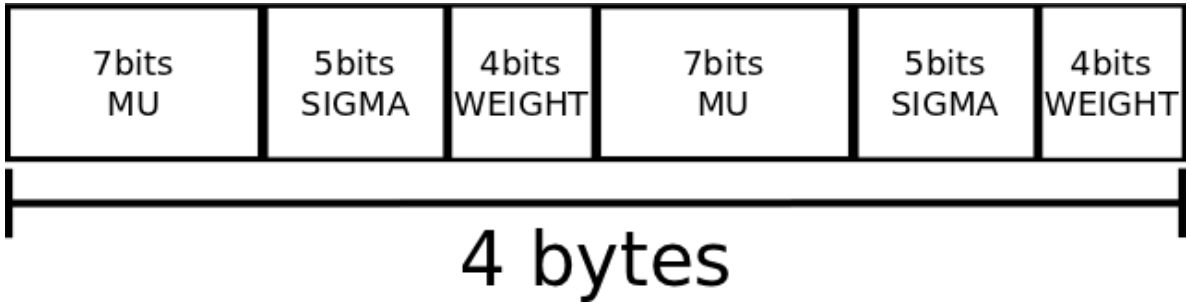
where $\mathbf{H}_n \in \{0, STEP\}$ is a set of independent binary random numbers, such that:

$$E\left[\sum_{n=1}^{A} \mathbf{H}_n\right] = L \tag{17}$$

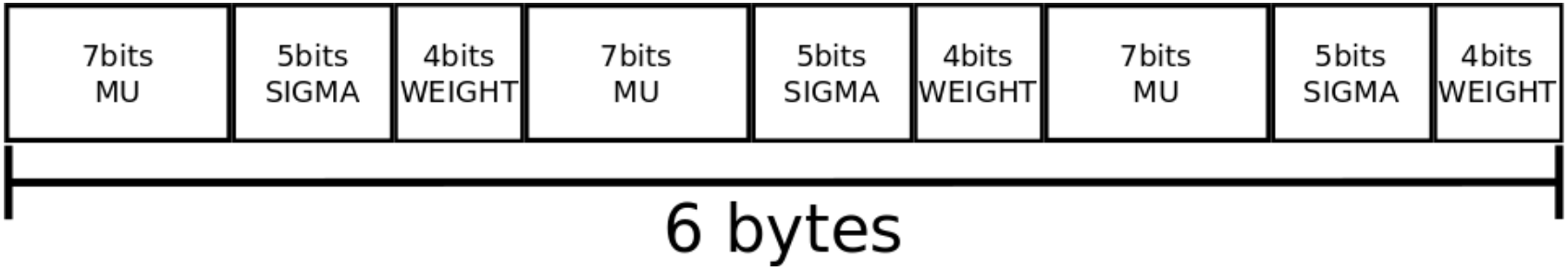
iGMM-I

We represent each Gaussian using 2 bytes

- 2 Gaussians:



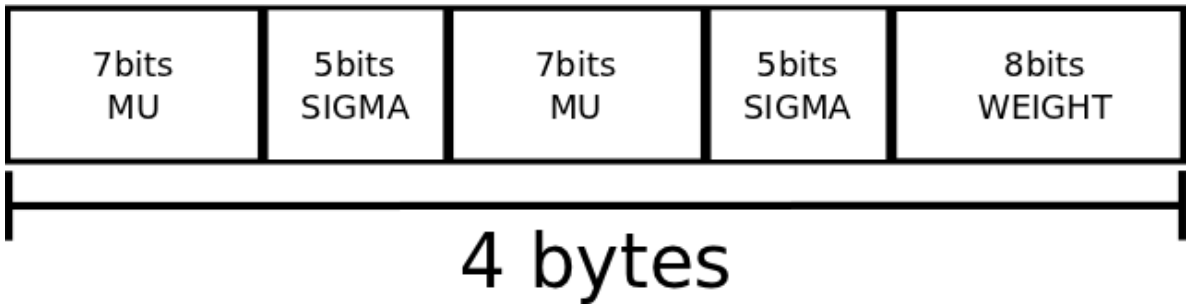
- 3 Gaussians:



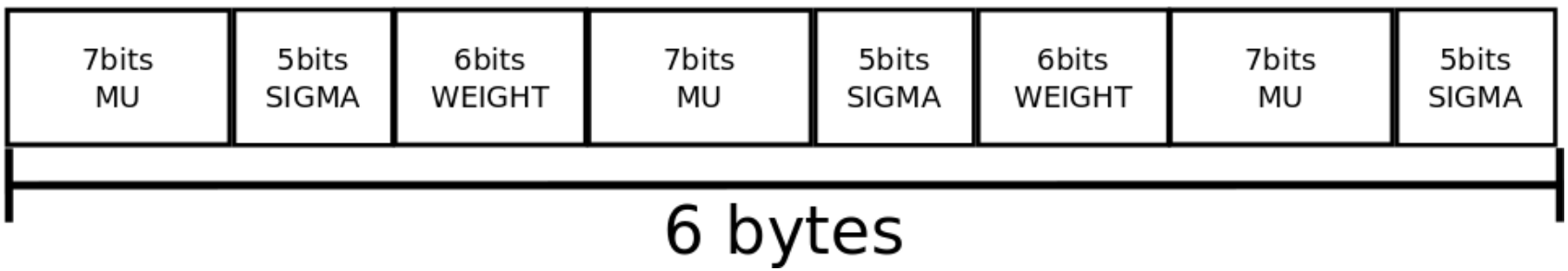
iGMM-s

We represent each Gaussian using 2 bytes

- 2 Gaussians:



- 3 Gaussians:



Note: To represent the G-th weight in iGMM-s the following relation is exploited: $\sum_{g=1}^{G} w_g = 1$

Memory requirements

Precision	Bytes per Gaussian	Footprint 2G (Bytes)	Footprint 3G (Bytes)
Double	24	921,600	1,382,400
Float	12	460,800	691,200
uint8_t	3	115,200	172,800
iGMM-x	2	76,800	115,200

The memory footprint of the algorithm is compressed by a factor 12:

- Feasible in the SeedEye board
- Tested also on an other chip with the lack of FPU (the one embedded into the Raspberry Pi)

Performance evaluation

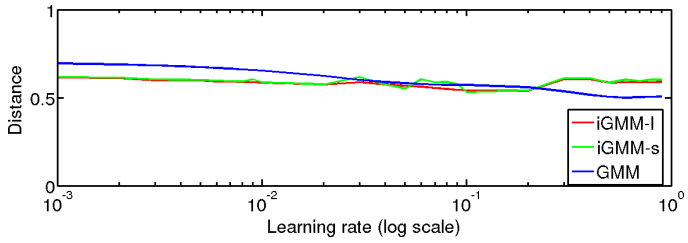
- The performance are evaluated using two standard data-sets which comprise different kind of movement:
  - Slow movement: the "IXMAS data-set"
  - Fast movement: the "Fudan Pedestrian data-set"
- Qualitative performance:
  - Based on Precision-Recall (or P-R) curves
  - The minimum distance of the P-R curves from the perfect classification point is computed for each value of learning rate so that the smaller the distance, the better the performance
- Quantitative performance:
  - Processing time comparison on SeedEye and Raspberry Pi boards

Performance evaluation

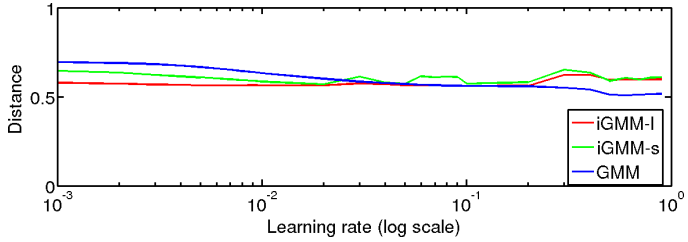
The minimum distance of the P-R curves from the perfect classification point

Fudan pedestrian datasets:

- 2 Gaussians:

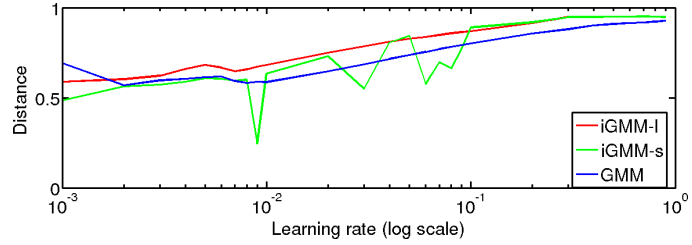


- 3 Gaussians:

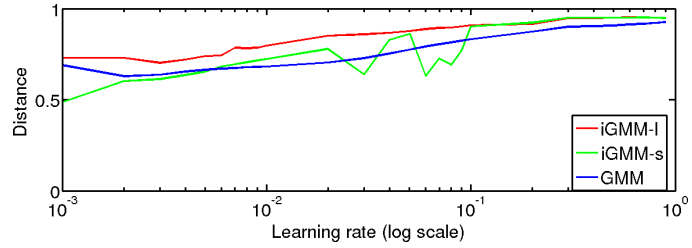


IXMAS:

- 2 Gaussians:

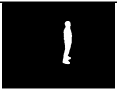
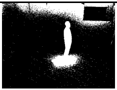
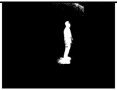
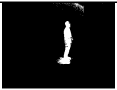
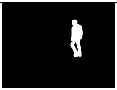
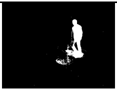
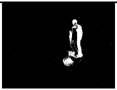
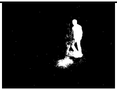
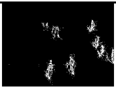
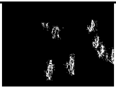


- 3 Gaussians:



Both iGMM-I and iGMM-s have comparable qualitative performance of GMM

Performance evaluation

Picture	Ground-truth	GMM	iGMM-l	iGMM-s
				
				
				
				

Performance evaluation

	GMM [s]	iGMM-I [s]	iGMM-s [s]
Rasp. Pi 2G QQ-VGA	0.033	0.007	0.010
Rasp. Pi 3G QQ-VGA	0.036	0.013	0.025
Rasp. Pi 2G Q-VGA	0.136	0.031	0.041
Rasp. Pi 3G Q-VGA	0.148	0.057	0.104
SeedEye 2G (40x40)	0.084	0.002	0.003
SeedEye 3G (40x40)	0.115	0.003	0.006

Retis
Real-Time Systems Laboratory

thank you

c.salvadori@sssup.it

References

References

M. Magrini, D. Moroni, C. Nastasi, P. Pagano, M. Petracca, G. Pieri, C. Salvadori, and O. Salvetti.
Visual sensor networks for infomobility.
Pattern Recognition and Image Analysis, 21:20–29, 2011.

M. Magrini, D. Moroni, P. Pagano, M. Petracca, G. Pieri, C. Salvadori, and Ovidio Salvetti.
Image mining for informability.
In *3rd Int. Workshop on Image Mining Theory and Applications (INSTICC Press, Angers, 2010)*, pages 35–44, Angers, France, 2010. INSTICC Press.

L. Maggiani, C. Salvadori, M. Petracca, P. Pagano, and Roberto Saletti.
Reconfigurable fpga architecture for computer vision applications in smart camera networks.
In *Distributed Smart Cameras (ICDSC), 2013 Seventh International Conference on*, 2013.

P. Pagano, M. Petracca, D. Alessandrelli, and C. Salvadori.
Is ict mature for an eu-wide intelligent transport system?
Intelligent Transport Systems, IET, 7(1):151–159, 2013.

M. Petracca, P. Pagano, Riccardo Pelliccia, Marco Ghibaudi, Claudio Salvadori, and Christian Nastasi.
On board unit hardware and software design for vehicular ad-hoc networks.
In R. Daher and A. Vinel, editors, *Roadside Networks for Vehicular Communications: Architectures, Applications and Test Fields*. IGI Global Press, 2012.

References

P. Pagano, C. Salvadori, S. Madeo, M. Petracca, S. Bocchino, D. Alessandrelli, A. Azzar, M. Ghibaudi, G. Pellerano, and R. Pelliccia.
A middleware of things for supporting distributed vision applications.
In *Proceedings of Workshop on Smart Cameras for Robotic Applications*, June 2012.

C. Salvadori, M. Petracca, M. Ghibaudi, and P. Pagano.
On-board Image Processing in Wireless Multimedia Sensor Networks: a Parking Space Monitoring Solution for Intelligent Transportation Systems.
CRC Press, December 2012.

M. Chitnis, C. Salvadori, M. Petracca, G. Lipari, and P. Pagano.
Traffic related observations by line sensing techniques.
In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 373–374, New York, NY, USA, 2010. ACM.

M. Chitnis, C. Salvadori, M. Petracca, P. Pagano, G. Lipari, and L. Santinelli.
Distributed visual surveillance with resource constrained embedded systems.
In Li-Minn Ang and Kah Phooi Seng, editors, *Visual Information Processing in Wireless Sensor Networks: Technology, Trends and Applications*. IGI Global Press, 2012.

M. Petracca, M. Ghibaudi, C. Salvadori, P. Pagano, and D. Munaretto.
Performance evaluation of FEC techniques based on BCH codes in video streaming over wireless sensor networks.

In *Proceedings of IEEE Symposium on Computers and Communications*, pages 43–48, Corfu, Greece, July 2011.

References

C. Salvadori, M. Petracca, R. Pelliccia, M. Ghibaudi, and P. Pagano.

Video streaming in wireless sensor networks with low-complexity change detection enforcement.
In *Proceedings of Baltic Congress on Future Internet Communications*, pages 8–13, Vilnius, Lithuania, April 2012.

C. Salvadori, M. Petracca, S. Madeo, S. Bocchino, and P. Pagano.

Video streaming applications in wireless camera networks: A change detection based approach targeted to 6lowpan.
Journal of Systems Architecture, (0):–, 2013.

C. Salvadori, D. Makris, M. Petracca, J. Martínez del Rincón, and S. A. Velastin.

Gaussian mixture background modelling optimisation for micro-controllers.
In *Advances in Visual Computing*, volume 7431 of *Lecture Notes in Computer Science*, pages 241–251. Springer Berlin Heidelberg, 2012.

Parking spaces monitoring (Solution 1)

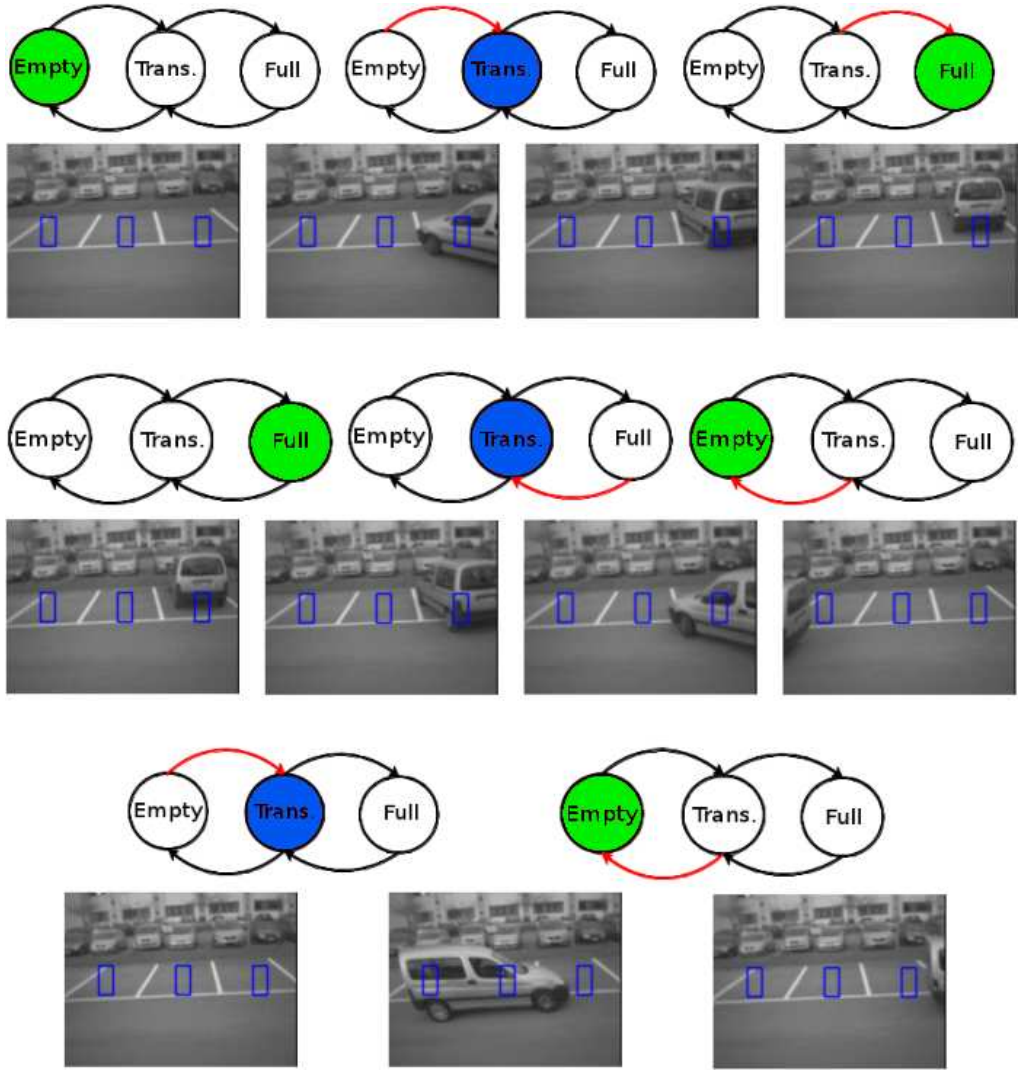
A multi-tier approach [MMN+11, MMP+10]:

- Fast detection algorithm:
  - Background subtraction
  - Histogram comparison
- Viola-Jones method:
  - Vector of Haar-like features are extracted from the parking lots RoIs
  - Cascade of binary classifiers

Performance summary:

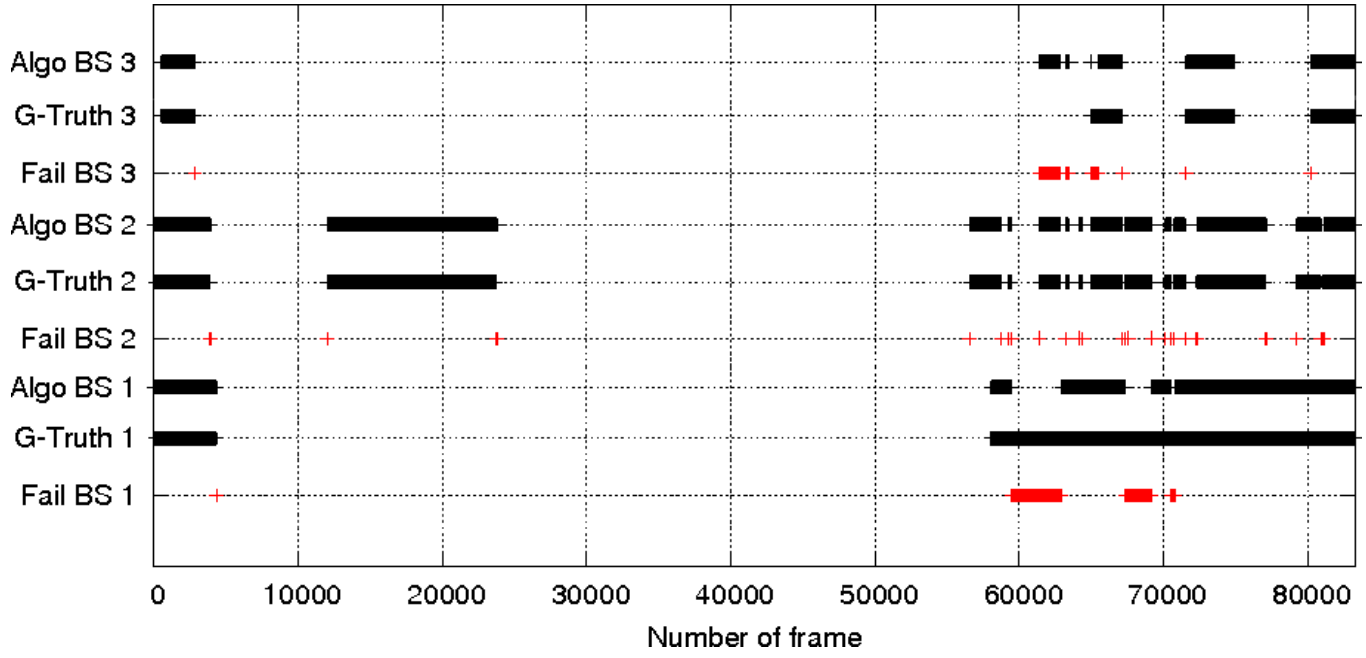
	Detection rate [%]	False alarm rate [%]
A classifier	99.5	50.0
The cascade (10 stages)	95.0	0.1

Parking spaces monitoring (Solution 2)



Parking spaces monitoring (Solution 2)

Performance summary:



Detection rate [%]	False alarm rate [%]
98.9	1.2

The vehicles counter application

Single Metric Model or SMM:

- The following two conditions have to be held:
  1. only a single event occurs in the monitored region (holistic approach);
  2. the size of the object have to be comparable with the monitored region size.
- Selected metric:

$$d(i) = \sum_{j=1}^{N} |p_{i,j} - b_j| \tag{18}$$

- Because the noise generated by the camera circuitry, $d(i)$ is modeled as a Gaussian $g_c(i)$ and updated using the following rules:

$$g_c(i) : \begin{cases} \mu_c(i) = (1-\delta) \cdot \mu_c(i-1) + \delta \cdot d(i) \\ \sigma_c(i) = (1-\delta) \cdot \sigma_c(i-1) + \delta \cdot (d(i) - \mu_c(i)) \end{cases} \tag{19}$$

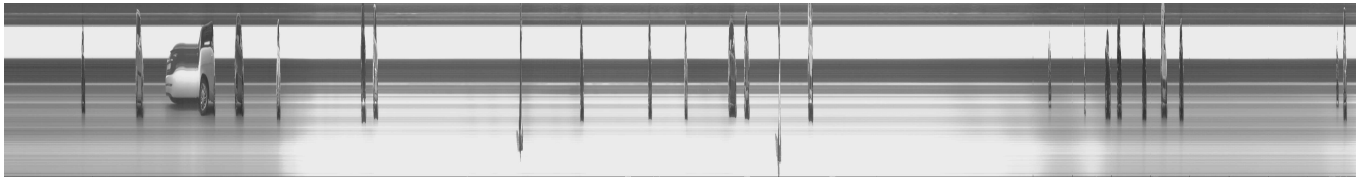
The vehicles counter application

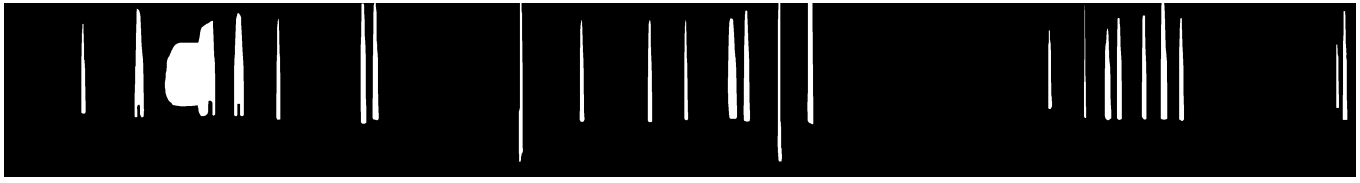
Thus, the background modeling is based on the comparison between the Gaussian $g_c(i)$ and the reference one g_r (constant illumination and no events), generating 3 events:

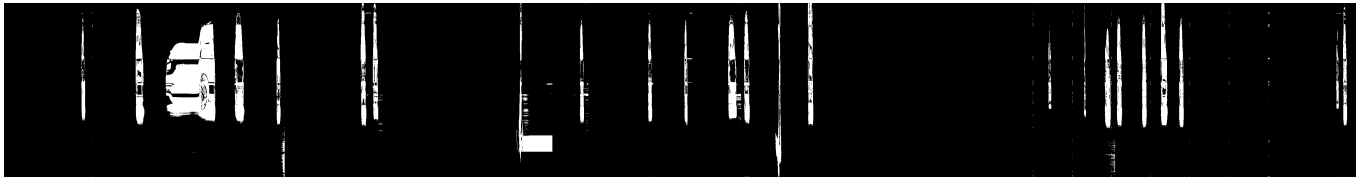
1. the *background event*, triggered when a sensitive variation in the mean value and a negligible one in the standard deviation occur. In this case the reference image b is overwritten by the current line (background update)
2. the *blob event*, triggered when a sensitive variation in both mean and standard deviation values occur. In this case the presence of an intraframe blob is validated counting the number of the foreground connected pixels
3. *no event*, when no meaningful variation occurs and no action is taken to save energy

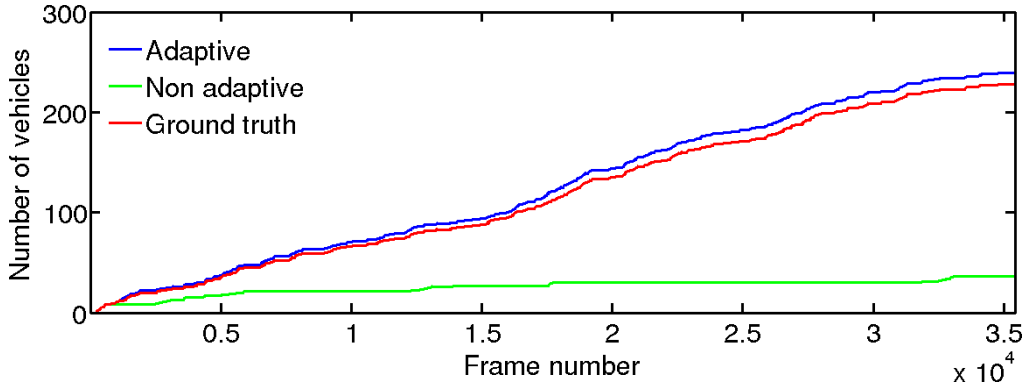
The vehicles counter application

Performance summary:









Video surveillance advanced applications

Performance summary:

(a) Ref. image.

(b) No Protection.

(c) $BCH(127, 113, 2)$.

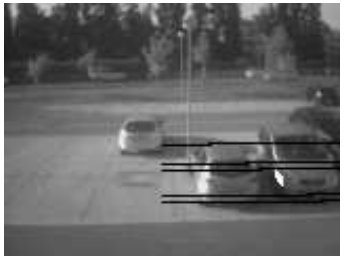
	No protection	$BCH(127, 113, 2)$
PRNR [dB]	56.8	61
Overhead [%]	0	22.5

DRAWBACK: only a video-stream can be instantiated!

Video surveillance advanced applications

Performance summary:

(a) Ref. image.



(b) Black concealment.

(c) Copy-bgnd concealment.

(d) Copy-image concealment.

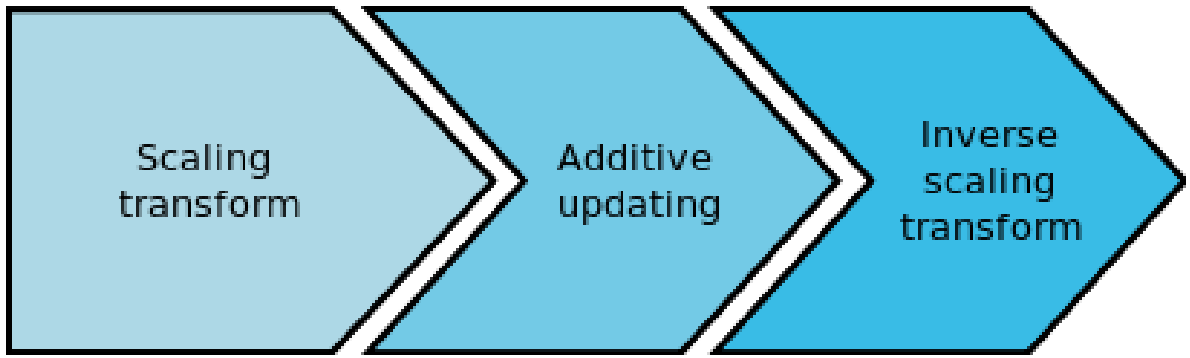
	Copy-image	Copy-bgnd	JPEG
PRNR [dB]	33.39	33.24	31.21
Data rate [Kbps]	40.04	40.04	40.78

At least 4 streams can be instantiated @1fps!

Mean value and variance updating

Updating methodology:

- Data are defined in the integer space $\mathbb{V}$ (in this case the domain represented in memory)
- Data are transformed to another space $\mathbb{U}$ (by using the scaling transform) where it is simpler to process
- Data are updated using a *generalised additive updating function* based on a *generalised-round* operator
- Data are converted back to $\mathbb{V}$ (with scaling inverse transform)



The scaling transform and its inverse

- Scaling transform $\chi(\circ) : \mathbb{V} \rightarrow \mathbb{U}$:

$$\chi(V) = V \cdot P \tag{20}$$

- Inverse scaling transform $\chi^{-1}(\circ) : \mathbb{U} \rightarrow \mathbb{V}$:

$$\chi^{-1}(E) = \frac{E}{P} \tag{21}$$

- P is the accuracy parameter

Generalised-round operator "G_ROUND"

The generalised-round operator is define as:

$$G\_ROUND(\delta, \xi, \gamma) = \begin{cases} \lfloor \delta \rfloor_\gamma & \text{if } |\delta| \geq \gamma \\ \gamma & \text{if } (|\delta| < \gamma) \wedge (|\delta| \geq \xi) \\ 0 & \text{if } (|\delta| < \gamma) \wedge (|\delta| < \xi) \end{cases} \tag{22}$$

where $\delta \in \mathbb{Z}$ is the value to be rounded

Generalised additive updating function

- We define *additive updating operations* all the functions $U : \mathbb{R} \to \mathbb{D}$ (in this case $\mathbb{D} \subseteq \mathbb{R}$) such that:

$$a(j+1) = U(a(j), \delta(j)) = a(j) + \delta(j) \tag{23}$$

  where $\delta(j)$ is called *updating contribution*

- Both the mean value and the variance updating rules are defined as *additive updating operations*:

$$\mu(j+1) = \mu(j) + \delta_\mu(d(j)) \text{ where } \delta_\mu(d(j)) = k(j)d(j) \tag{24}$$

$$\sigma^2(j+1) = \sigma^2(j) + \delta_{\sigma^2}(d^2(j))$$
$$\text{where } \delta_{\sigma^2}(d^2(j)) = k(j)(d^2(j) - \sigma^2(j)) \tag{25}$$

Generalised additive updating function

The *generalised additive updating function* $\bar{U}$ is defined in the integer space $\mathbb{U}$ ($\bar{U}(\circ) : \mathbb{U} \rightarrow \mathbb{U}$) taking into account the *generalised-round* operator, as follows:

$$\bar{U}(a(j), \delta(j)) = G\_ROUND[\chi[a(j+1)]] = \\ = \chi[a(j)] + \{G\_ROUND[\chi[\delta(j)]]\} \quad (26)$$

Because of the linearity of all the involved operators, the overall result of all the chain can be summarized by the following relation:

$$a(j+1) = a(j) + \chi^{-1}\{G\_ROUND[\chi[\delta(j)]]\} \quad (27)$$

Calculating the updating step ξ

Because of the membership criterion, both the updating contributions are bounden inside the *updating contribution ranges*:

$$\delta^{\mu}_{min}(\sigma^2) = -k\sigma\sqrt{T} \leq \delta_{\mu}(d_{i,j}) \leq k\sigma\sqrt{T} = \delta^{\mu}_{MAX}(\sigma^2) \quad (28)$$

$$\delta^{\sigma^2}_{min}(\sigma^2) = -k\sigma^2 \leq \delta_{\sigma^2}(d^2_{i,j}) \leq k\sigma^2(T-1) = \delta^{\sigma^2}_{MAX}(\sigma^2) \quad (29)$$

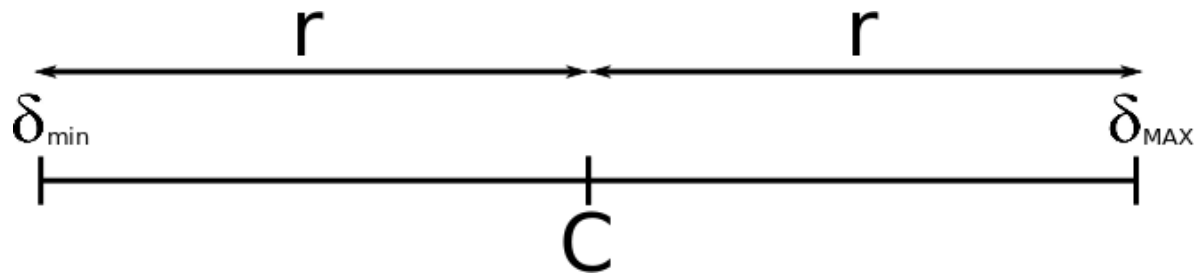
We can represent these ranges using the following parameters:

- The *center* C

$$C = \frac{\delta_{MAX} + \delta_{min}}{2} \quad (30)$$

- The *radius* r

$$r = \frac{\delta_{MAX} - \delta_{min}}{2} \quad (31)$$



Calculating the updating step ξ

Both these parameters depends on the variance σ^2

$C_\mu(\sigma^2) = 0$	$r_\mu(\sigma^2) = \alpha\sigma^2\sqrt{T}$
$C_{\sigma^2}(\sigma^2) = \frac{\alpha T\sigma^2 - 2\alpha\sigma^2}{2}$	$r_{\sigma^2}(\sigma^2) = \frac{\alpha\sigma^2 T}{2}$

Because of the bounded nature of the variance:

$$C_\mu = 0 \qquad \forall\sigma^2 \tag{32}$$

$$r_\mu \in [r_\mu(\sigma^2_{min}), r_\mu(\sigma^2_{MAX})] \tag{33}$$

$$C_{\sigma^2} \in [C_{\sigma^2}(\sigma^2_{min}), C_{\sigma^2}(\sigma^2_{MAX})] \tag{34}$$

$$r_{\sigma^2} \in [r_{\sigma^2}(\sigma^2_{min}), r_{\sigma^2}(\sigma^2_{MAX})] \tag{35}$$

Calculating the updating step ξ

The idea is to define *critical intervals* close to the border of both the above mentioned ranges, where the updating contribution is considered sufficiently large to increase the counter:

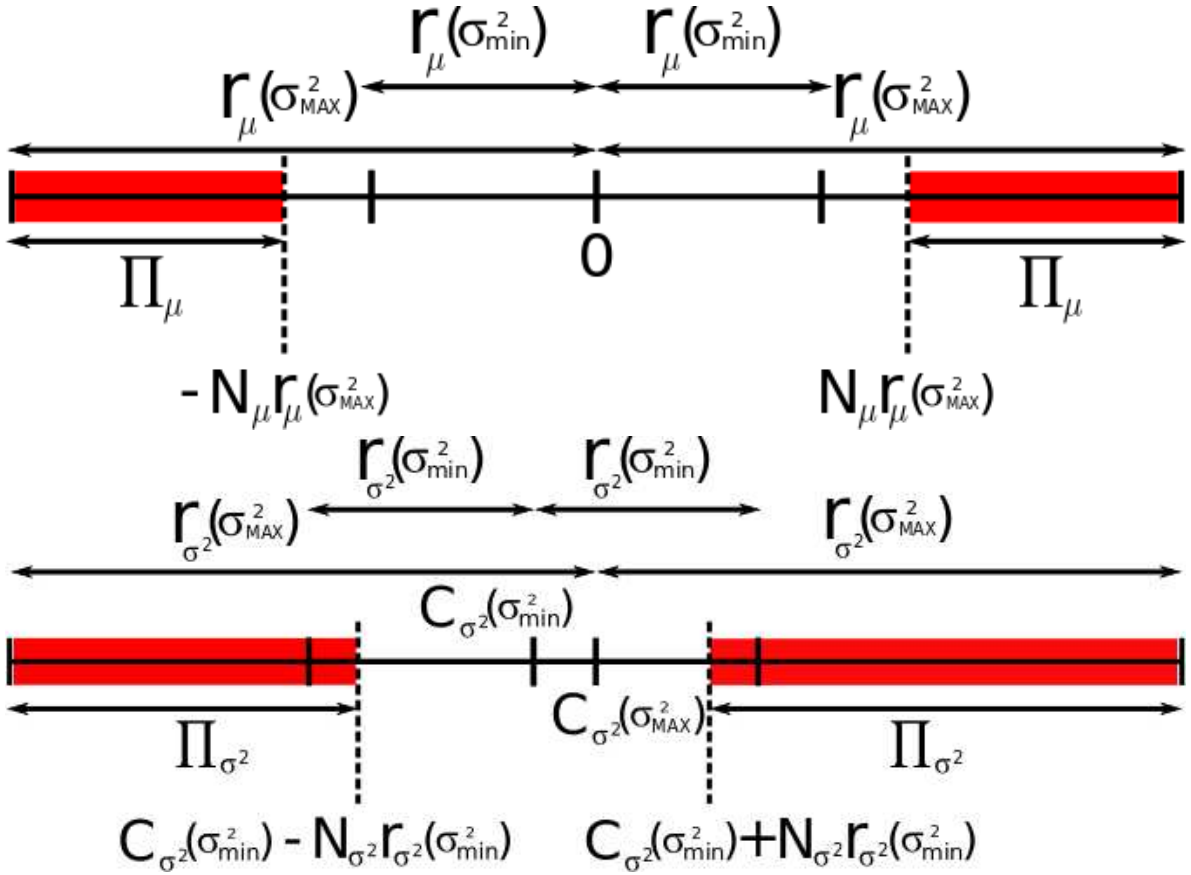
- The mean value depends directly on the difference value $d_{i,g}(j)$ (where $d_{i,g}(j) \in \mathbb{Z}$)
- The mean value depends directly on the square of the difference value $d_{i,g}(j)$ (where $d_{i,g}(j) \in \mathbb{Z}$)

In this situation we can define:

- the critical interval Π_μ is the zone inside the maximum width membership interval where the updating contribution δ_μ is considered sufficiently large to update the mean value of its granularity
- the critical interval Π_{σ^2} is the zone inside the minimum width membership interval where the updating contribution δ_{σ^2} is considered sufficiently large to update the variance of its granularity

Calculating the updating step ξ

The critical intervals are defined using the parameters $N_\mu \in \mathbb{R}$ and $N_{\sigma^2} \in \mathbb{R}$ (with $N_\mu \in (0,1)$ and $N_{\sigma^2} \in (0,1)$)



Calculating the updating step ξ

In mathematical terms:

$$\Pi_\mu = \{\delta_\mu \in \mathbb{Z} : |\delta_\mu - C_\mu| \geq \lfloor N_\mu r_\mu(\sigma^2_{MAX}) \rfloor\} \quad (36)$$

$$\Pi_{\sigma^2} = \{\delta_{\sigma^2} \in \mathbb{Z} : |\delta_{\sigma^2} - C_{\sigma^2}| \geq \lfloor N_{\sigma^2} r_{\sigma^2}(\sigma^2_{min}) \rfloor\} \quad (37)$$

... and then the the updating steps can be computed as:

$$\xi_\mu = N_\mu \cdot r_\mu(\sigma^2_{MAX}) \quad (38)$$

$$\xi_{\sigma^2} = N_{\sigma^2} \cdot r_{\sigma^2}(\sigma^2_{min}) \quad (39)$$

The values of N_μ and N_{σ^2} are calibrated iteratively to minimize the difference of the iGMM-x and GMM intervals on a validation data-set

Stochastic weights updating

- From discrete random number theory and because of the linearity of the expected value $E[\circ]$

$$E\left[\sum_{n=1}^{A} \mathbf{H}_n\right] = STEP \cdot \sum_{n=1}^{A} P_{STEP}(n) = L \tag{40}$$

- $P_{STEP}(n)$ is known and derives from the chosen weight approximation
- The idea is to implement the logic making use of:
  1. The updating probability $P_{STEP}(n)$
  2. A uniformly distributed random number $\mathbf{X}_n$ defined in the interval $[0, MAX]$
  3. A threshold T_X

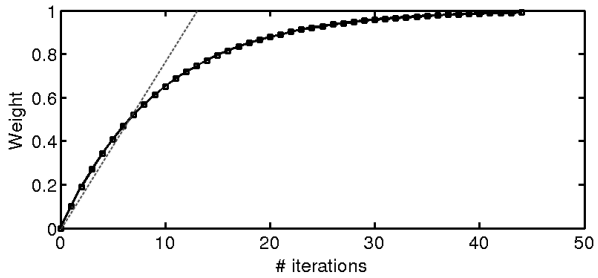
$$P_{STEP}(n) = P(\mathbf{X}_n > T_X) \tag{41}$$

- The threshold T_X is computed as:

$$T_X = [1 - P_{STEP}(n)] \cdot MAX \tag{42}$$

iGMM-I: Linear approximation

$$w(s) = s\frac{w_0}{s(w_0, \alpha)} \qquad (43)$$



- the stochastic updating method described is applied
- Because of the linear approximation, the probability $P_{STEP}(n)$ to update the weight (counter) is constant and independent to n:

$$P_{STEP}(\alpha) = \frac{L}{STEP \cdot A} \qquad (44)$$

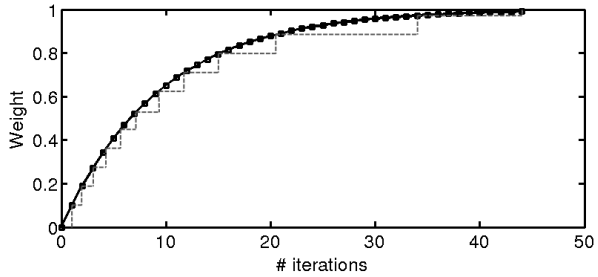
iGMM-s: Staircase approximation

All the possible weight values may be represented by an exponential distribution as defined that has been uniformly sampled over the iterations axis

- Sampling period ν (weight domain):

$$\nu = \frac{1}{L} \tag{45}$$

  where $L = 2^l - 1$



- Stochastic updating is the core mechanism

The updating probability is defined as how many iterations are compressed at every counter updating of the value *STEP*:

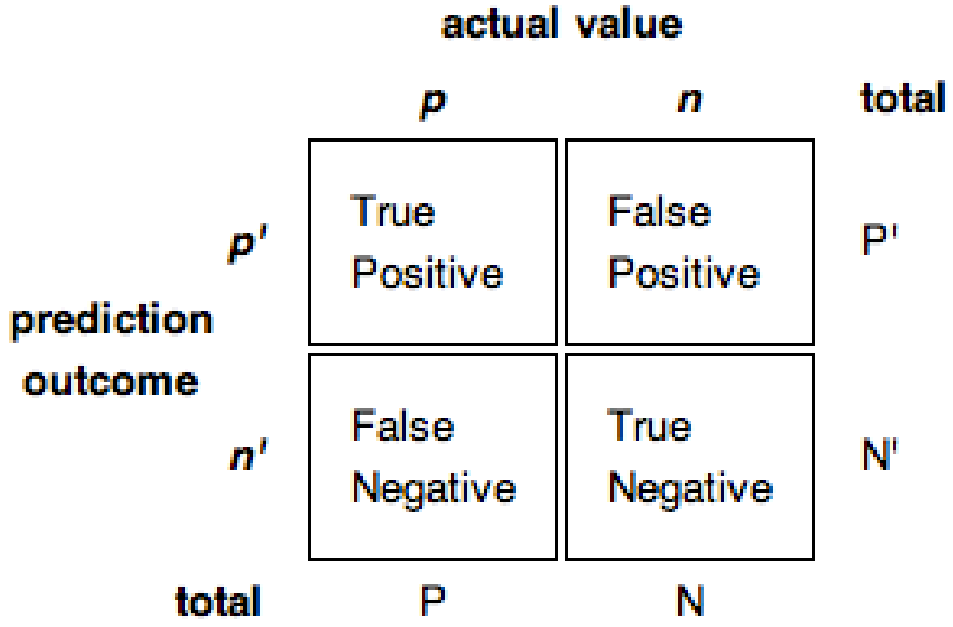
$$P_{STEP}(c, \alpha) = \frac{STEP}{\bar{\nu}(c, \alpha)} \tag{46}$$

iGMM-s: Staircase approximation

A *Look-Up Table* (or *LUT*) is instantiated inside a vector of L elements to contain the thresholds computed as a function of $P_{STEP}(c, \alpha)$ These assumptions are valid only for the increasing trend curve, but:

- The increasing curve is symmetric to the decreasing curve
- Consequently:
  1. if the pixel p_i belongs to the current Gaussian, the counter c is increased by $STEP$ if and only if the random value $\mathbf{V} > LUT_T(\alpha)[c]$;
  2. if the pixel p_i does not belong to the current Gaussian, the counter c is decreased by $STEP$ if and only if the random value $\mathbf{V} > LUT_T(\alpha)[\bar{S} - c]$, where $\bar{S}$ is the number of iterations needed to reach the weight value 0.99 from 0;
  3. otherwise no operation is performed

P-R curves



$$P = \frac{TP}{TP + FP} \tag{47}$$

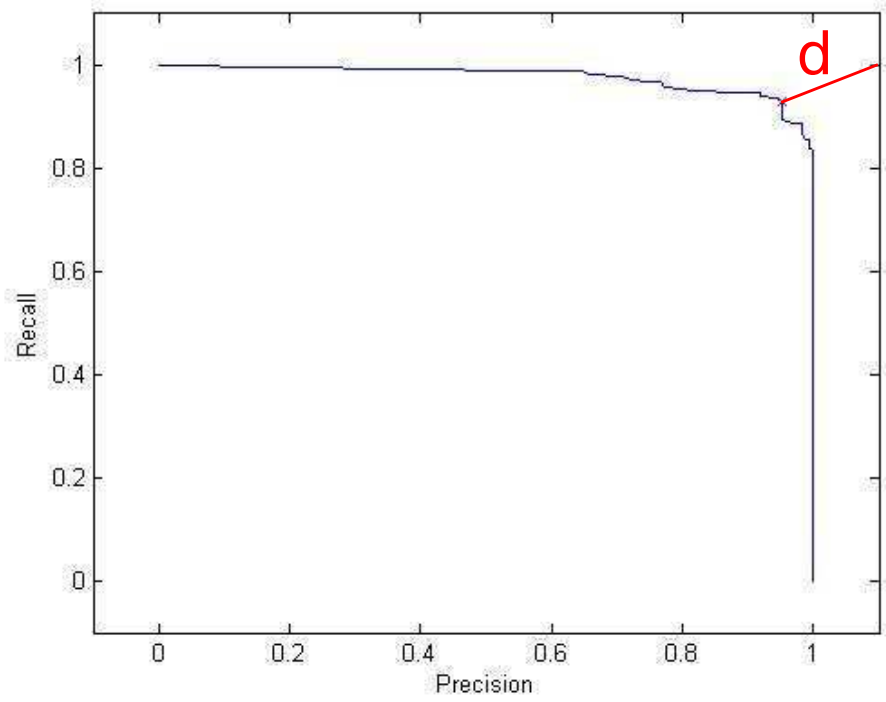
$$R = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{48}$$

Perfect classification: $(P, R) = (1, 1)$

P-R curves

P-R curves:

Fixing the learning rate α and varying the threshold T, the point $(P(T), R(T))$ is computed



Then, the minimum distance d from the perfect classification point is computed.