Talk:

CHANGE DETECTION ALGORITHMS

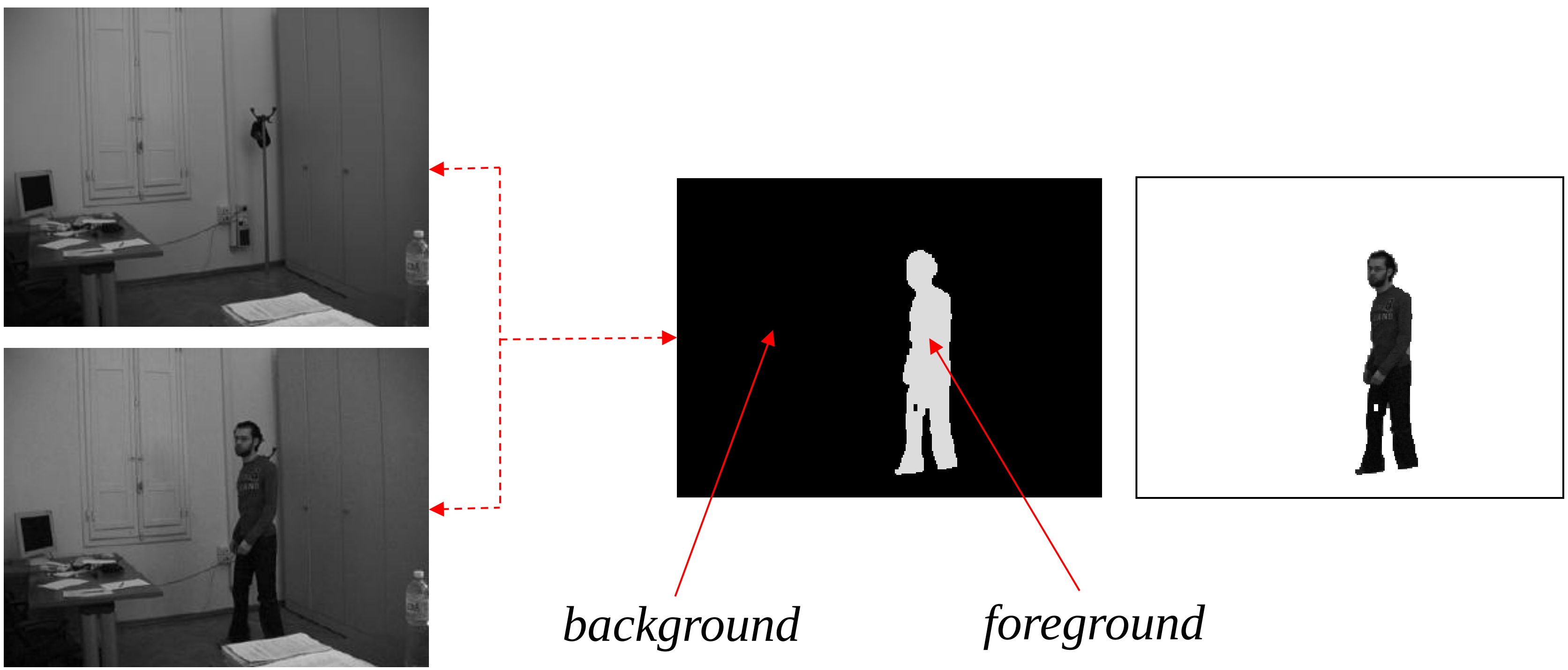
Bologna, November 29th, 2013

Dott. Ing. Alessandro Lanza
University of Bologna

• **Change Detection:** detection of “meaningful” changes occurring in a scene by processing of images captured at different time instants.

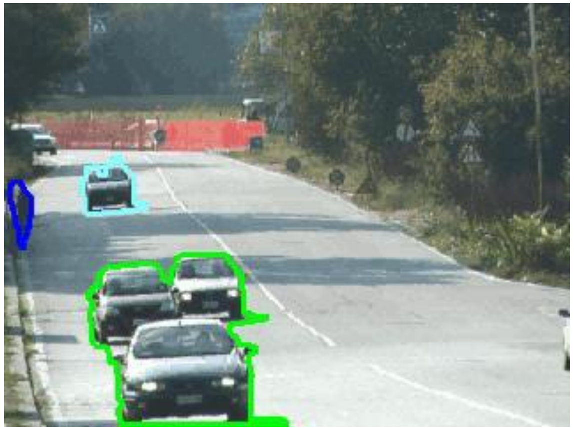
• **Input:** two (at least!) or more images of the monitored scene.

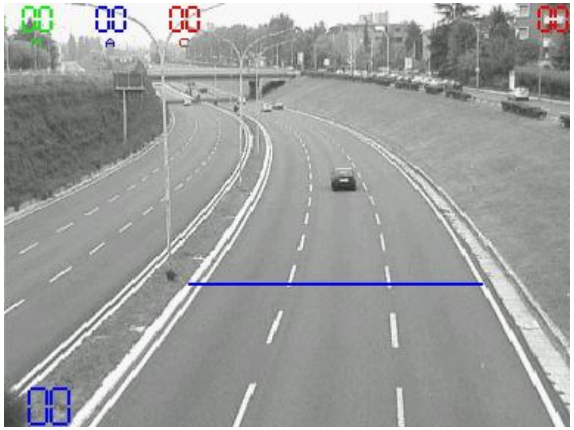
• **Output:** binary image, called "change mask“: each pixel is assigned one between two values (labels) *c*, *u* ("changed", "unchanged"): *c* if meaningful changes occur at the pixel, *u* otherwise (commonly, $c = 255$, $u = 0$ $\longrightarrow$ white/ black).



• **Videosurveillance**

• **Traffic monitoring**

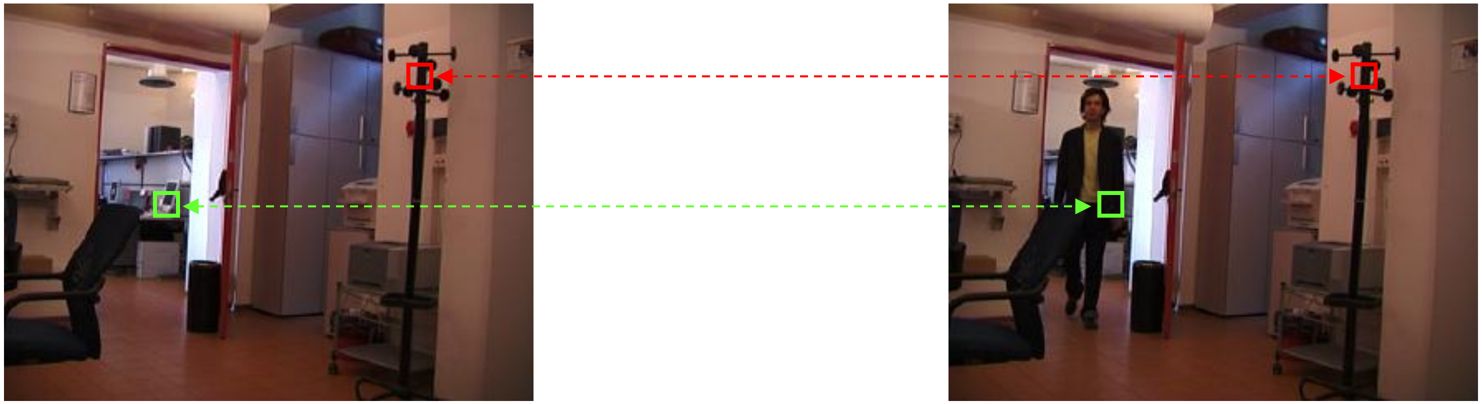




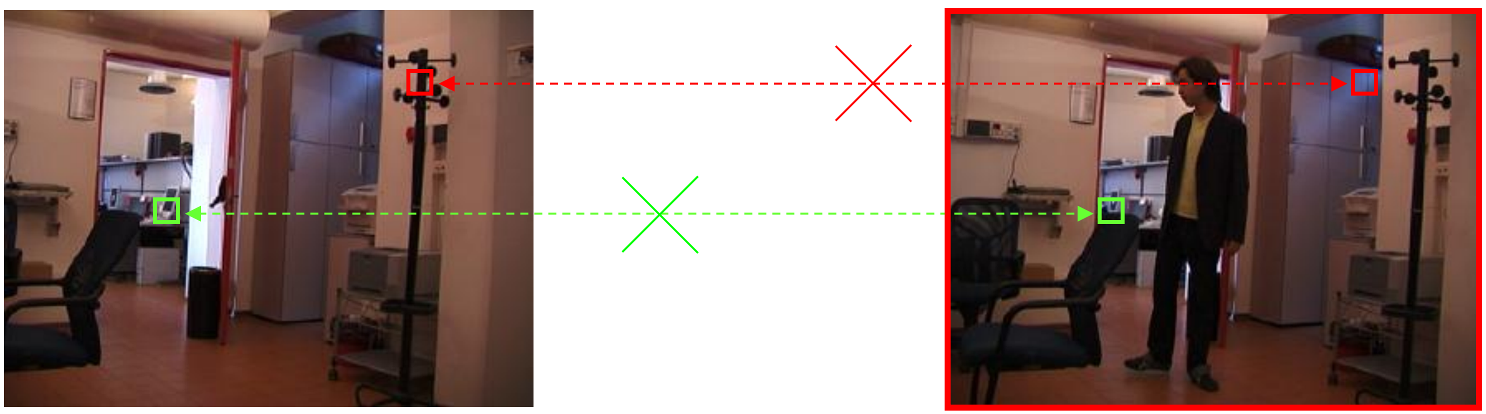
• **Video compression (MPEG)**

Change Detection Algorithms - **Working assumptions**

- **Static camera:** pixels having the same coordinates in the two images represent the same physical portion of scene surface → change detection by direct pixel by pixel comparison.

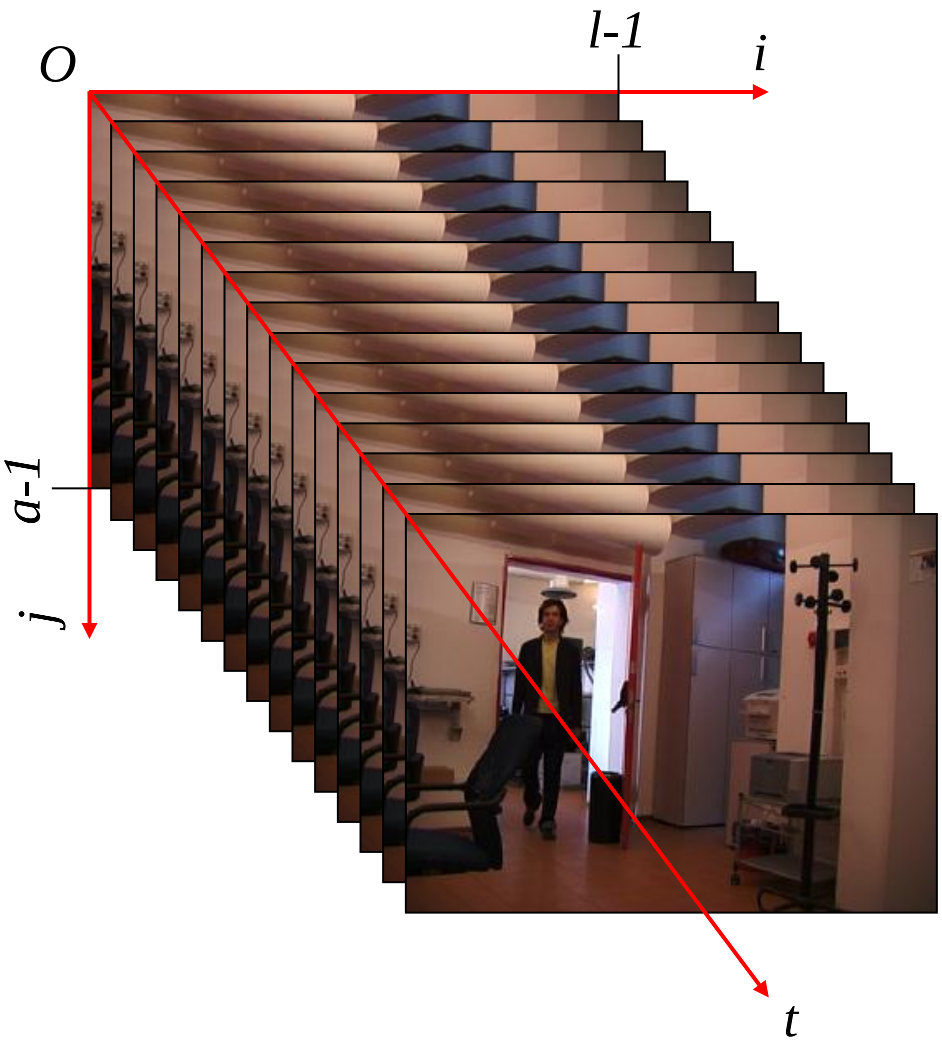


- **Moving camera:** change detection for static camera after geometric registration of the two views.



• **Low frame rate** (temporal sampling frequency of frames, frames/s): change detection by comparison between the current frame and one of the previous frames.

• **High frame rate** (tipically > 1 frames/s): change detection by comparison between the current frame and a set of previous frames (dynamics of changes can be exploited).



- **Single frame:**

number of channels

$$\vec{F}:(i,j)\in R^2 \mapsto \vec{F}(i,j)\in R^{n_c}$$

- Digitalization (spatial sampling + quantization):

$$\vec{F}:(i,j)\in Z^2 \mapsto \vec{F}(i,j)\in Z^{n_c}$$

- Notations:

width (pixels) height (pixels)

$$(i,j)\in[0,l-1]\times[0,a-1]$$

$$F_c(i,j)\in[0,p-1] \quad c=1,\dots,n_c$$

Depth (intensity levels)

- **Sequence of frames:**

$$S=\{\vec{F}_t, t\in R\}$$

- temporal sampling:

$$S=\{\vec{F}_t, t=0,1,\dots\}$$

- **based on "temporal frame difference":**

$$C_t = f\left(\vec{F}_t, \vec{F}_{t-1}, \ldots, \vec{F}_{t-k}\right)$$

''small''

- *two-frame difference (k = 1):*

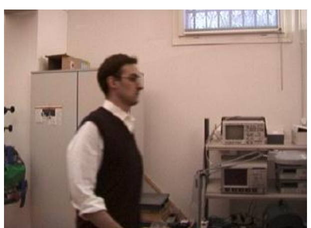
$\vec{F}_{t-1}$

$\vec{F}_t$

$$C_t = f\left(\vec{F}_t, \vec{F}_{t-1}\right)$$

- *three-frame difference (k = 2):*

$\vec{F}_{t-2}$



$\vec{F}_{t-1}$

$\vec{F}_t$

$$C_t = f\left(\vec{F}_t, \vec{F}_{t-1}, \vec{F}_{t-2}\right)$$

- **based on "background subtraction":**

$$C_t = f\left(\vec{F}_t, \vec{B}_t\right)$$

where

it can be "very big'

$$B_t = g\left(\vec{F}_{t-1}, \vec{F}_{t-2}, \ldots, \vec{F}_{t-k}\right)$$

represents the background of the monitored scene

$\vec{B}_t$

$\vec{F}_t$

$$C_t = f\left(\vec{F}_t, \vec{B}_t\right)$$

- *necessity of "background maintenance":*
  - *background modelling*
  - *background initialization*
  - *background differencing*
  - *background updating*

- sequence of 385 frames, sampled at 12,5 frames/s, 320x240 pixels, gray level and color (RGB):

- Salient features:
  - moving "objects" are poorly textured (small gradients of colors / gray levels);
  - "objects" moving with a wide range of velocities (from very fast to still);
  - initial subsequence free of moving "objects";
  - stationary illumination conditions;
  - static background;

distance image *distance function* $d : R^{n_c} \times R^{n_c} \mapsto R$

$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = d\left(\vec{F}_t(i,j), \vec{F}_{t-1}(i,j)\right) > T \\ 0 & \text{otherwise} \end{cases} \longrightarrow \textit{threshold}$$

• distance functions based on Hölder norms of the difference vector:

$$\vec{v} \in R^n \rightarrow \left\|\vec{v}\right\|_p = \sqrt[p]{\sum_{i=1}^{n} \left|v_i\right|^p} \qquad \overrightarrow{\Delta F}_t(i,j) = \vec{F}_t(i,j) - \vec{F}_{t-1}(i,j)$$

$$D_t(i,j) = d\left(\vec{F}_t(i,j), \vec{F}_{t-1}(i,j)\right) = \left\|\overrightarrow{\Delta F}_t(i,j)\right\|_p$$

• gray level images (one channel, $n_c = 1$):

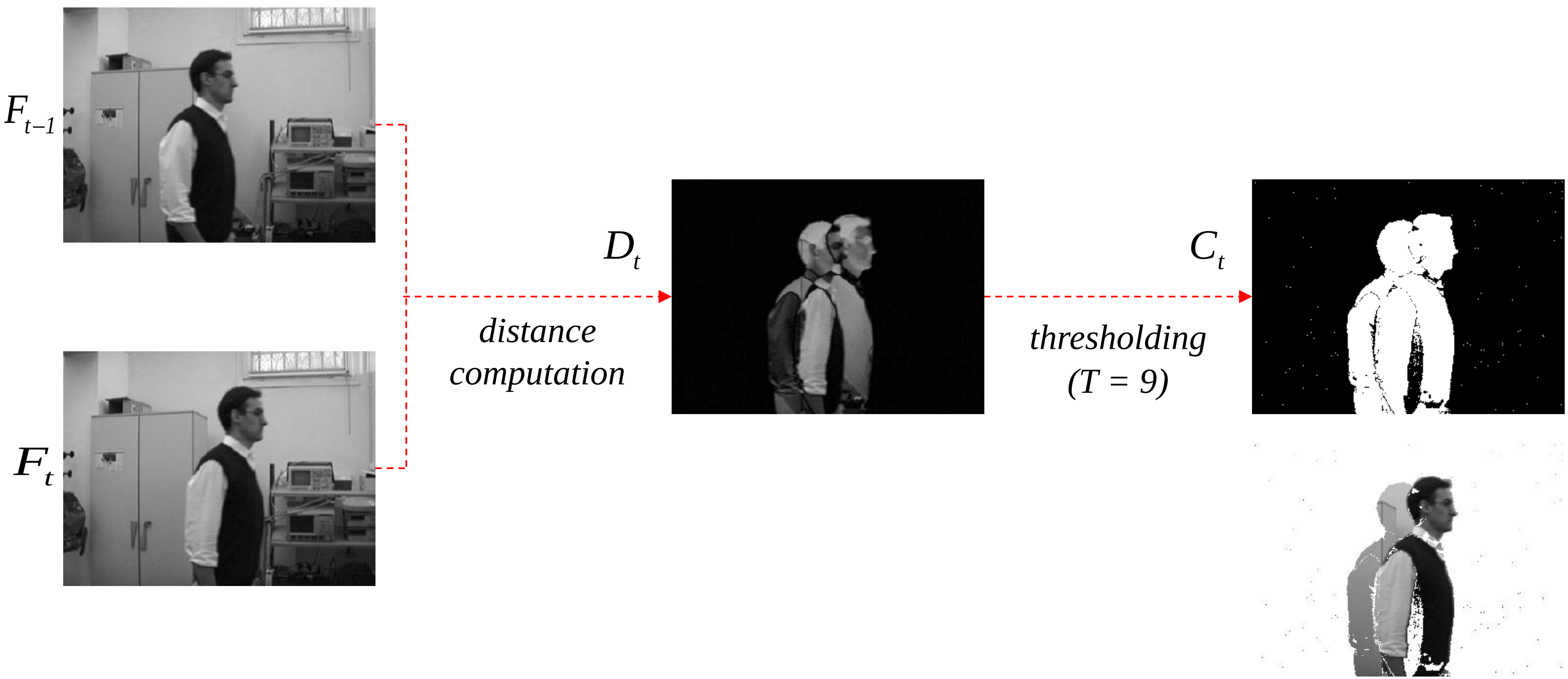
$$D_t(i,j) = \left|F_t(i,j) - F_{t-1}(i,j)\right| \quad \forall p$$

• color (RGB) images (three channels, $n_c = 3$):

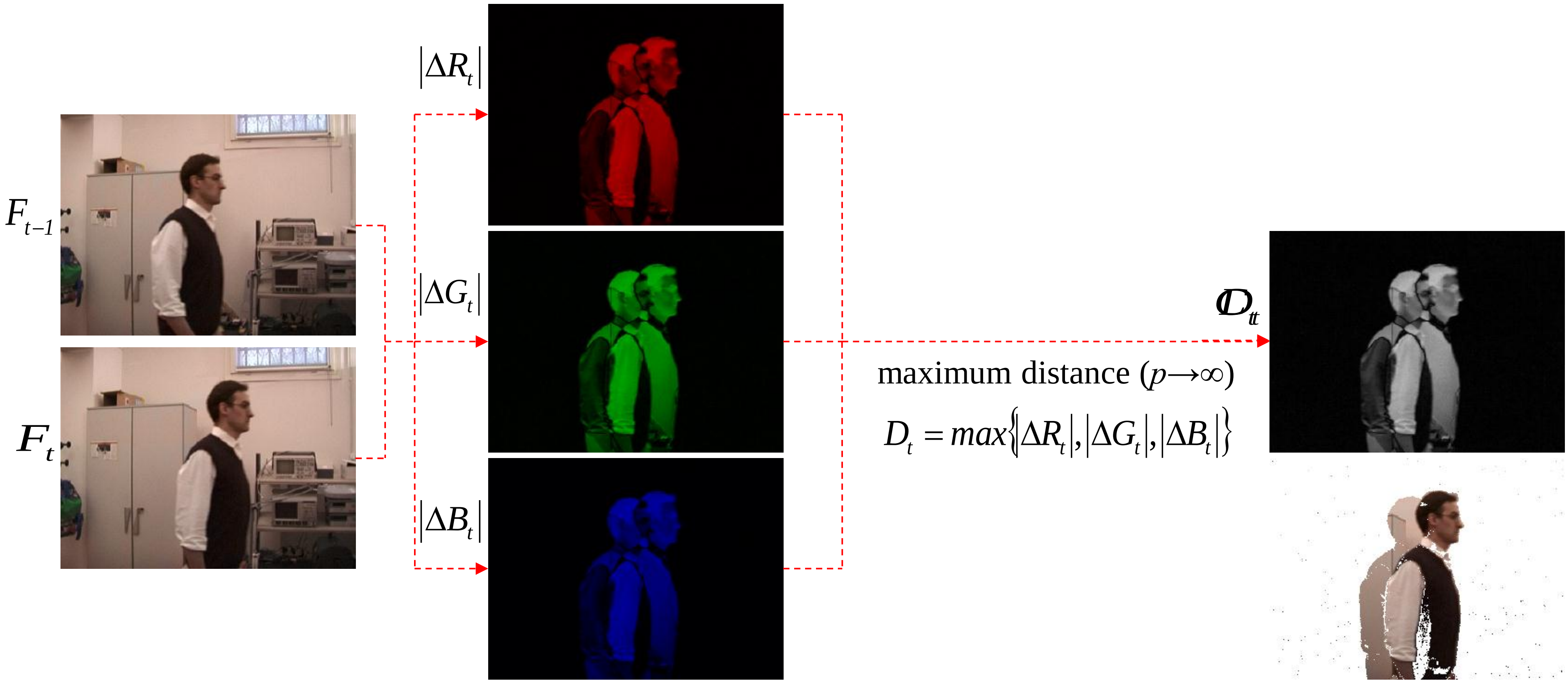
$$\vec{F}_{t-1}(i,j) = \begin{bmatrix} R_{t-1}(i,j) \\ G_{t-1}(i,j) \\ B_{t-1}(i,j) \end{bmatrix} \quad \vec{F}_t(i,j) = \begin{bmatrix} R_t(i,j) \\ G_t(i,j) \\ B_t(i,j) \end{bmatrix} \qquad \overrightarrow{\Delta F}_t(i,j) = \begin{bmatrix} \Delta R_t(i,j) = R_t(i,j) - R_{t-1}(i,j) \\ \Delta G_t(i,j) = G_t(i,j) - G_{t-1}(i,j) \\ \Delta B_t(i,j) = B_t(i,j) - B_{t-1}(i,j) \end{bmatrix}$$

$$D_t(i,j) = \sqrt[p]{\left|\Delta R_t(i,j)\right|^p + \left|\Delta G_t(i,j)\right|^p + \left|\Delta B_t(i,j)\right|^p}$$

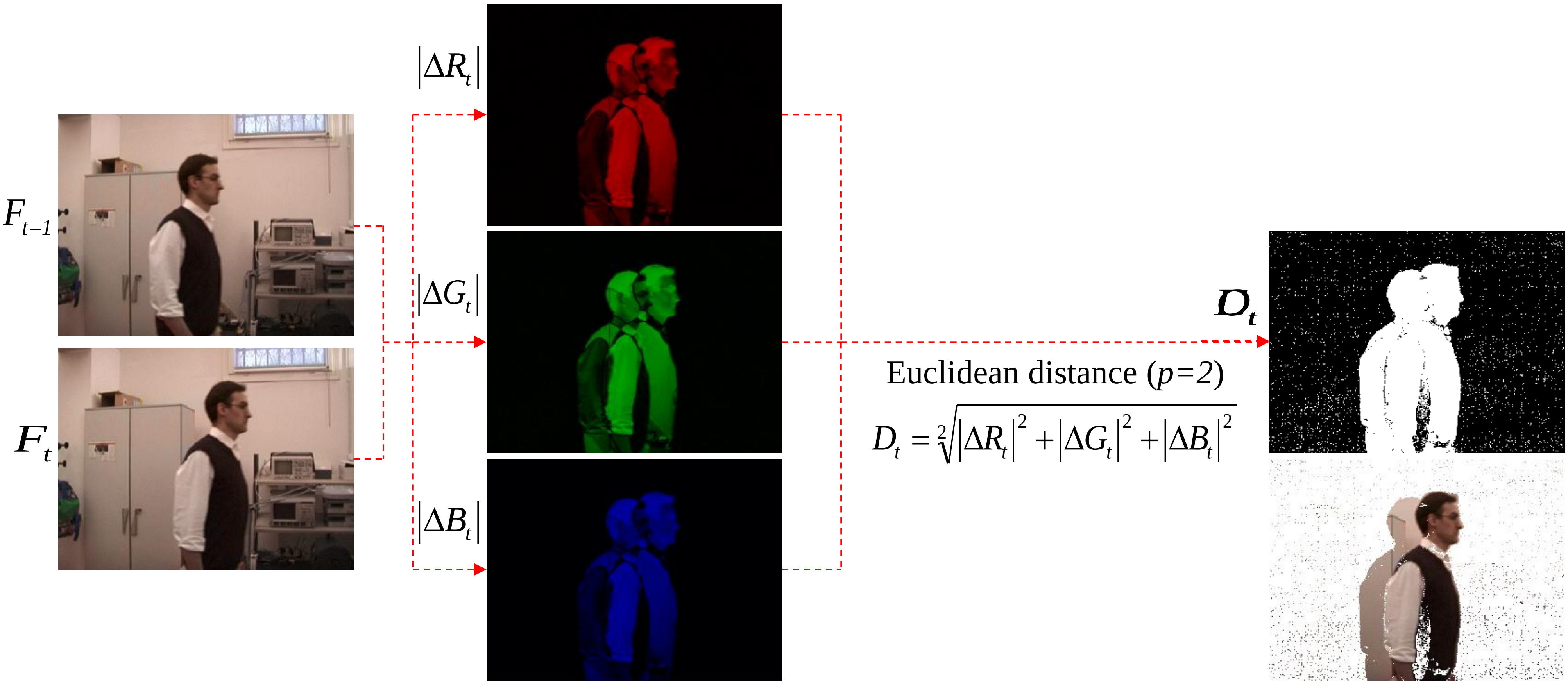
$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = \left| F_t(i,j) - F_{t-1}(i,j) \right| > T \\ 0 & \text{otherwise} \end{cases}$$



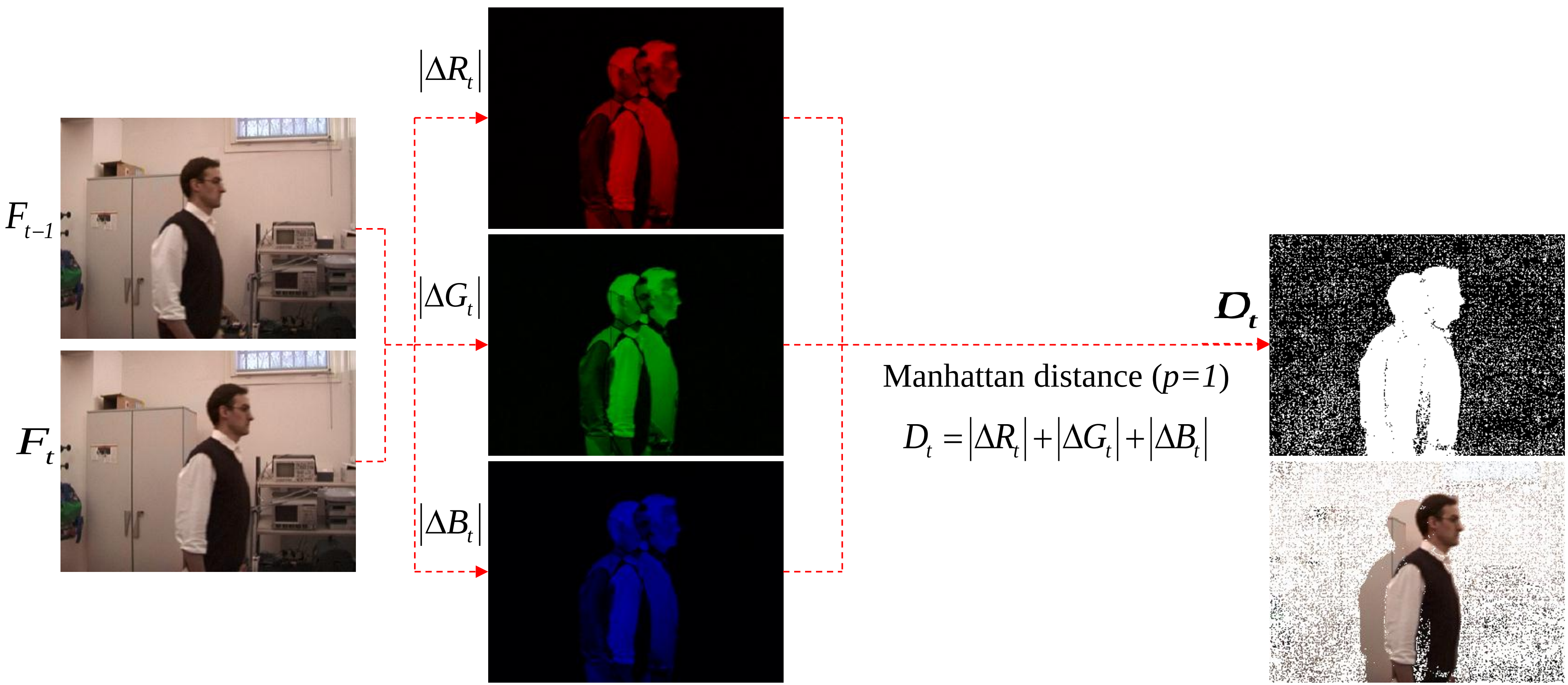
$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = \sqrt[p]{\left|\Delta R_t(i,j)\right|^p + \left|\Delta G_t(i,j)\right|^p + \left|\Delta B_t(i,j)\right|^p} > T \\ 0 & \text{otherwise} \end{cases}$$

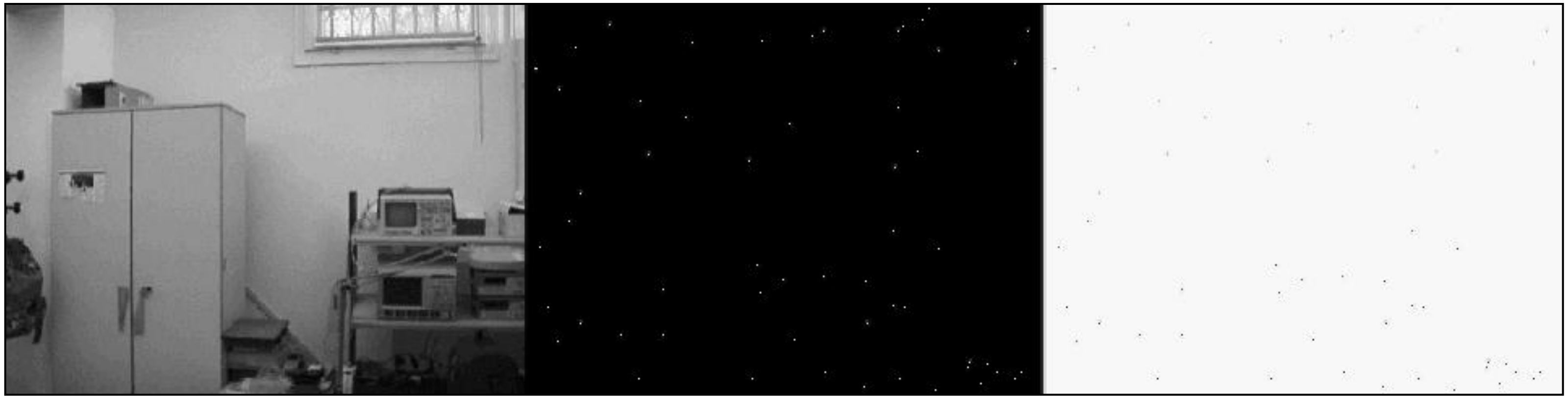


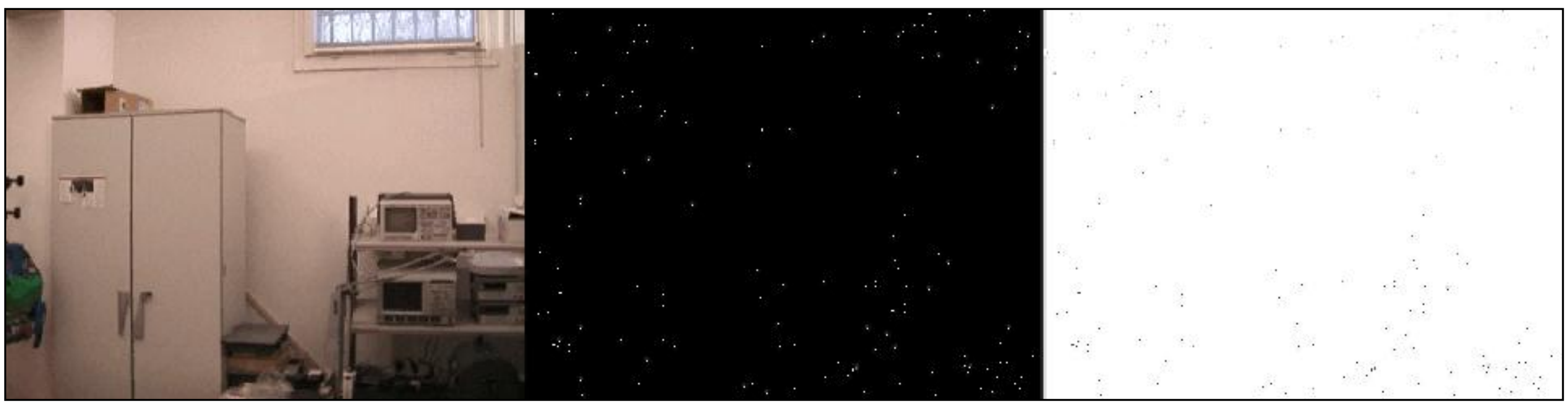
$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = \sqrt[p]{\left|\Delta R_t(i,j)\right|^p + \left|\Delta G_t(i,j)\right|^p + \left|\Delta B_t(i,j)\right|^p} > T \\ 0 & \text{otherwise} \end{cases}$$

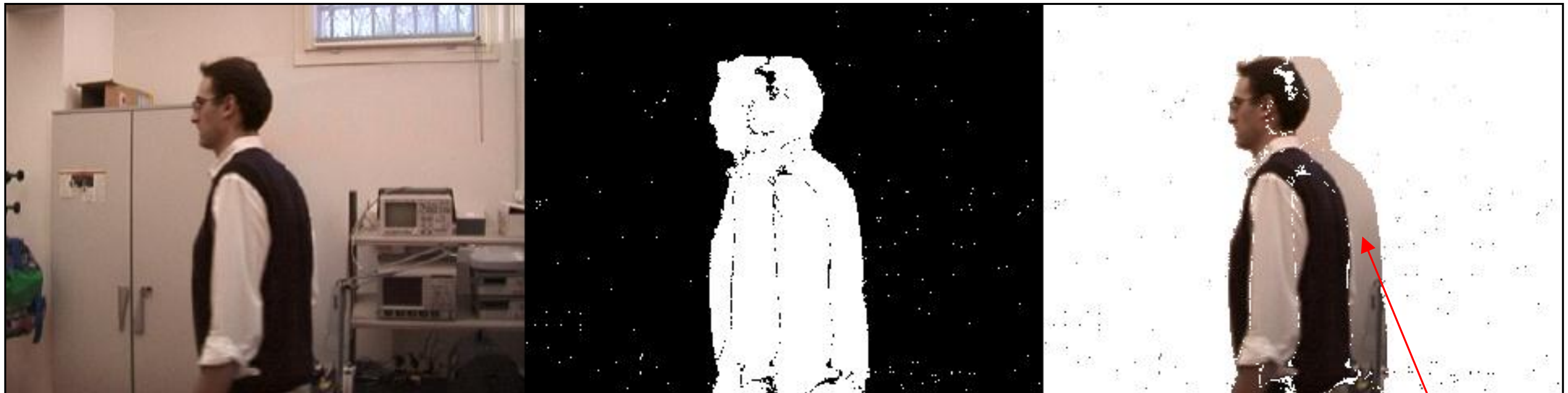


$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = \sqrt[p]{\left|\Delta R_t(i,j)\right|^p + \left|\Delta G_t(i,j)\right|^p + \left|\Delta B_t(i,j)\right|^p} > T \\ 0 & \text{otherwise} \end{cases}$$

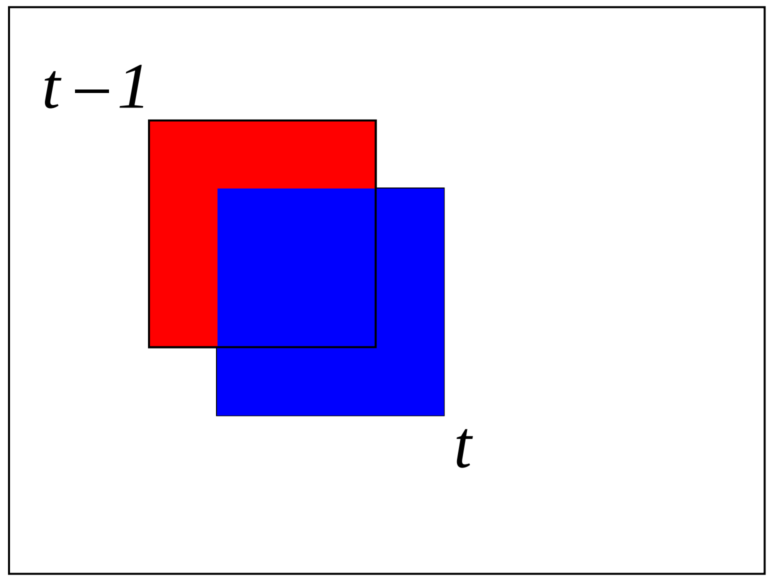




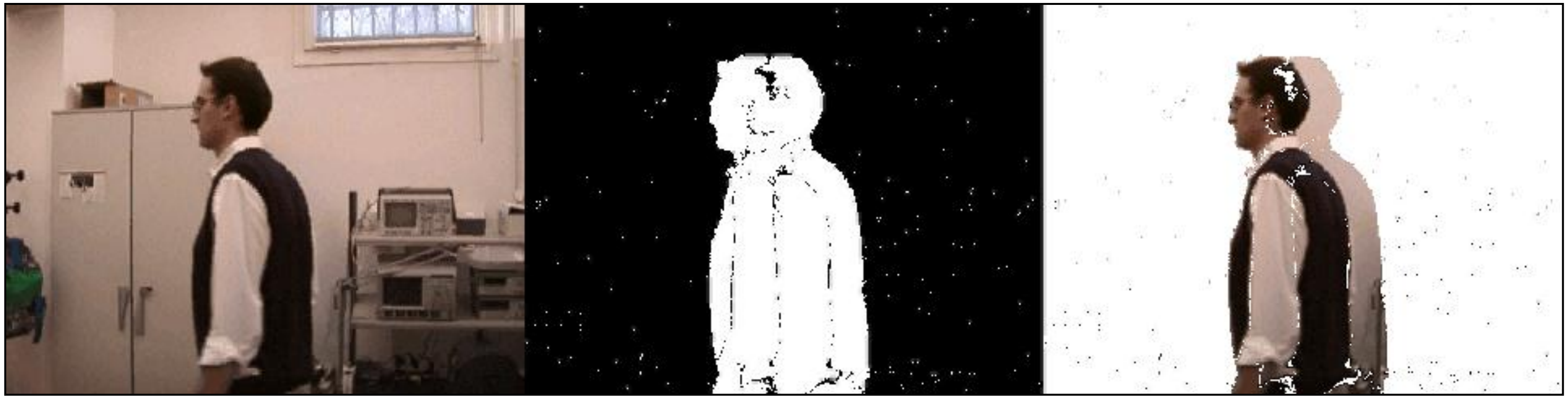




- *pixels detected as "changed"*
- *true positive pixels*
- *true negative pixels*
- *false positive pixels*
- *false negative pixels*



ghosting
(false positives):
the greater,
the faster the
moving objects



- *pixels detected as "changed"*
- *true positive pixels*
- *true negative pixels*
- *false positive pixels*
- *false negative pixels*

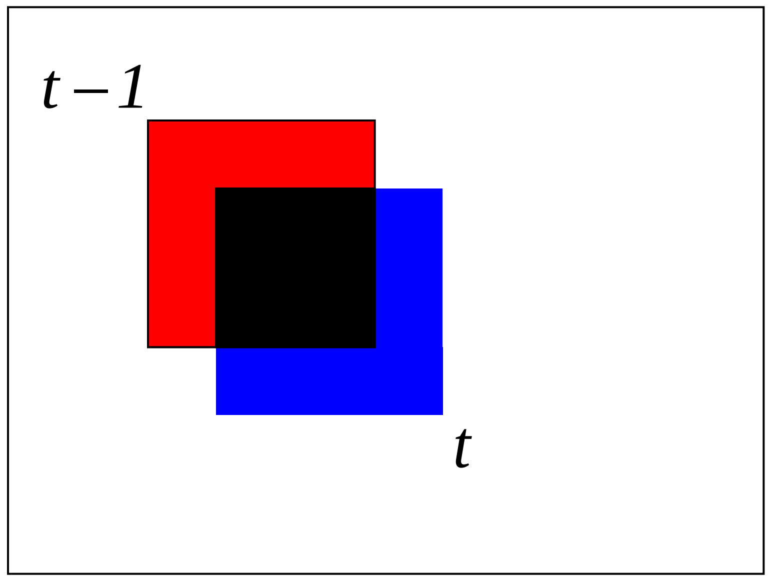
pixels detected as "changed"

true positive pixels

true negative pixels

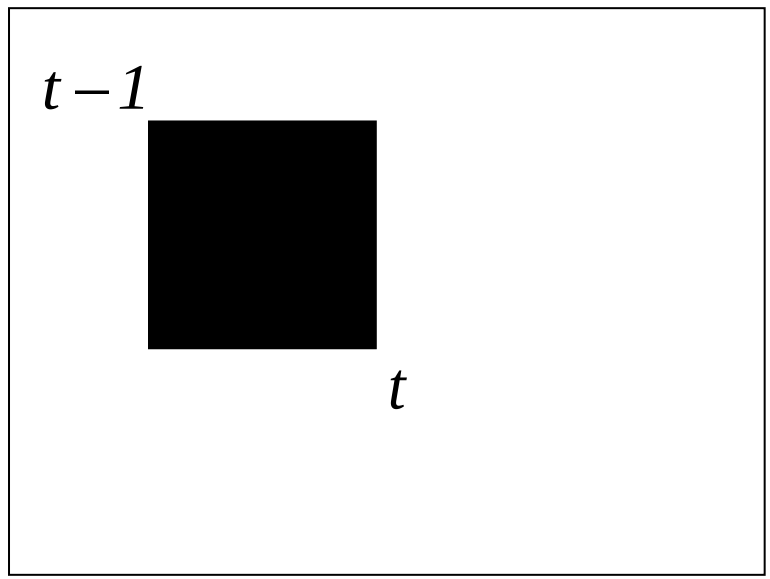
false positive pixels

false negative pixels



foreground aperture (false negatives): the greater, the less "textured" the moving objects

- *pixels detected as "changed"*
- *true positive pixels*
- *true negative pixels*
- *false positive pixels*
- *false negative pixels*



disappearance of stationary objects (false negatives): independent of objects textureness

- *pixels detected as "changed"*
- *true positive pixels*
- *true negative pixels*
- *false positive pixels*
- *false negative pixels*

• We perform twice the two-frame difference...between the current frame and the two most recent previous frames, separately, then we compute the intersection (binary AND) between the two obtained change masks:

$$C_t(i,j) = \begin{cases} 255 & if \quad d\left(\vec{F}_t(i,j), \vec{F}_{t-1}(i,j)\right) > T \quad and \quad d\left(\vec{F}_t(i,j), \vec{F}_{t-2}(i,j)\right) > T \\ 0 & otherwise \end{cases}$$

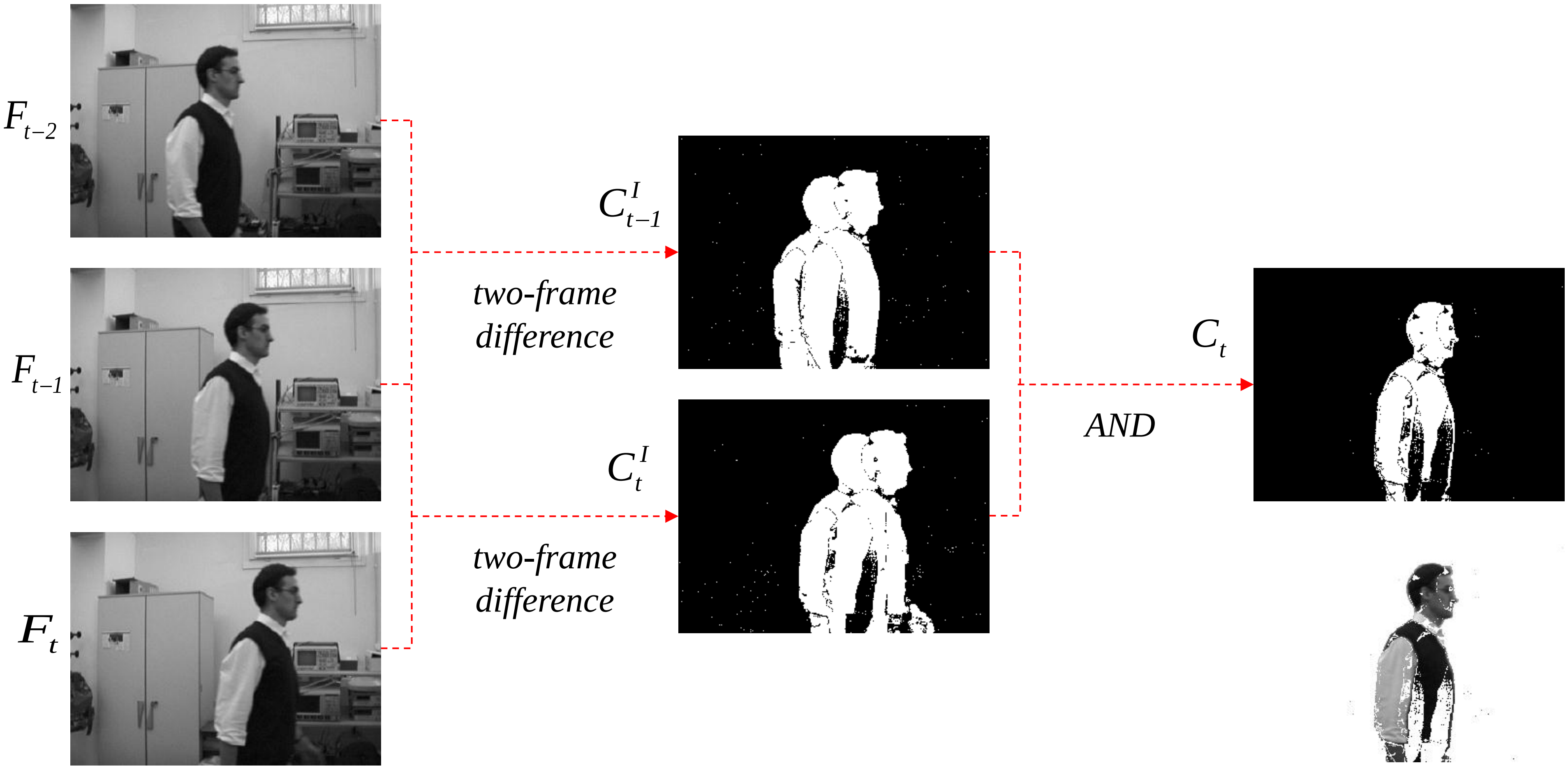
• admitting a one-frame delay in the change mask computation, it is better the following:

$$C_{t-1}(i,j) = \begin{cases} 255 & if \quad d\left(\vec{F}_t(i,j), \vec{F}_{t-1}(i,j)\right) > T \quad and \quad d\left(\vec{F}_{t-1}(i,j), \vec{F}_{t-2}(i,j)\right) > T \\ 0 & otherwise \end{cases}$$

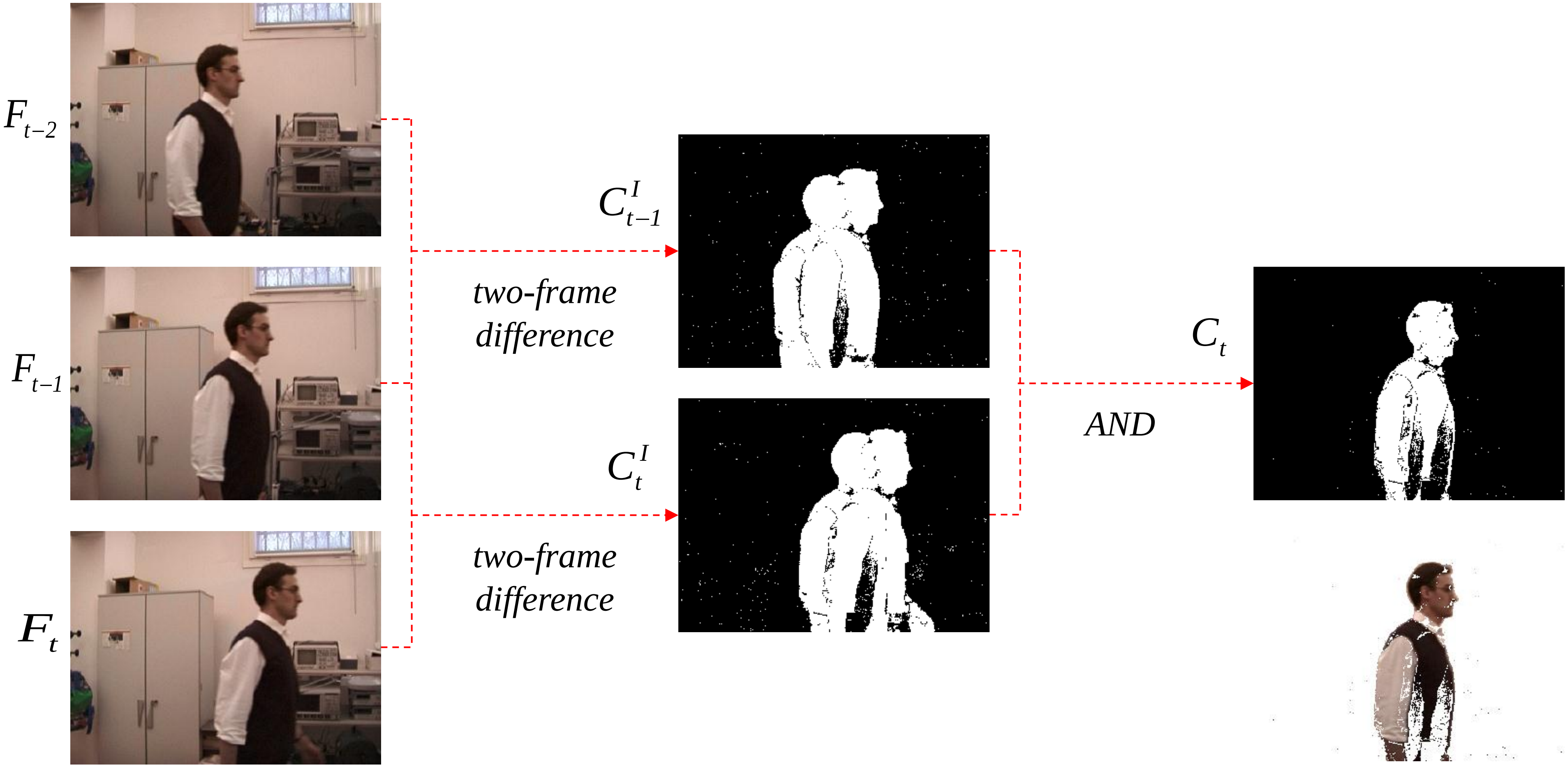
• that can be re-written in a more compact form as:
:

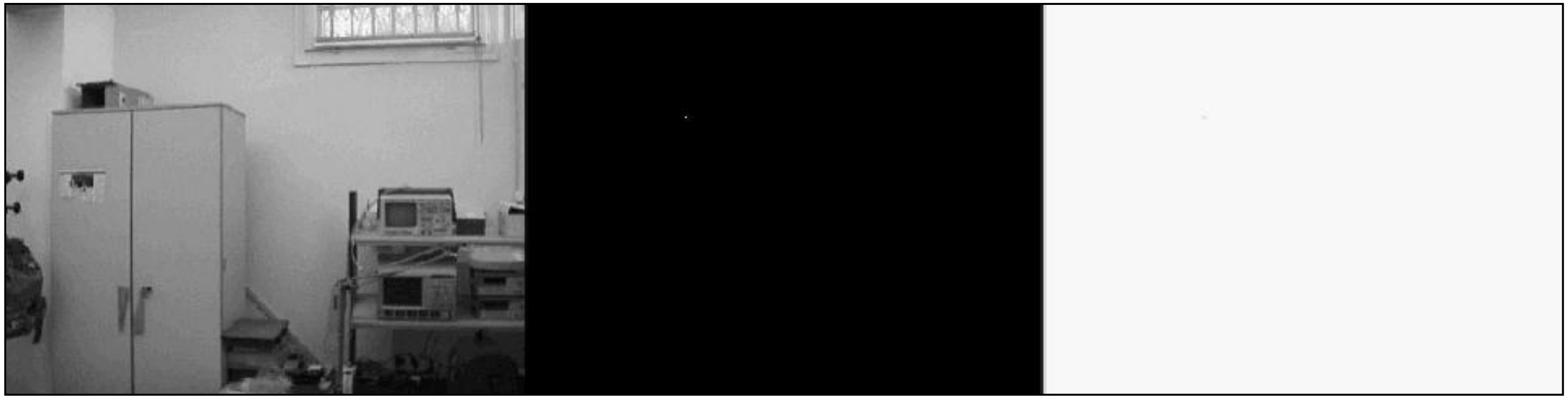
$$C_{t-1}(i,j) = \begin{cases} 255 & if \quad C_t^I(i,j) = 255 \quad and \quad C_{t-1}^I(i,j) = 255 \\ 0 & otherwise \end{cases}$$

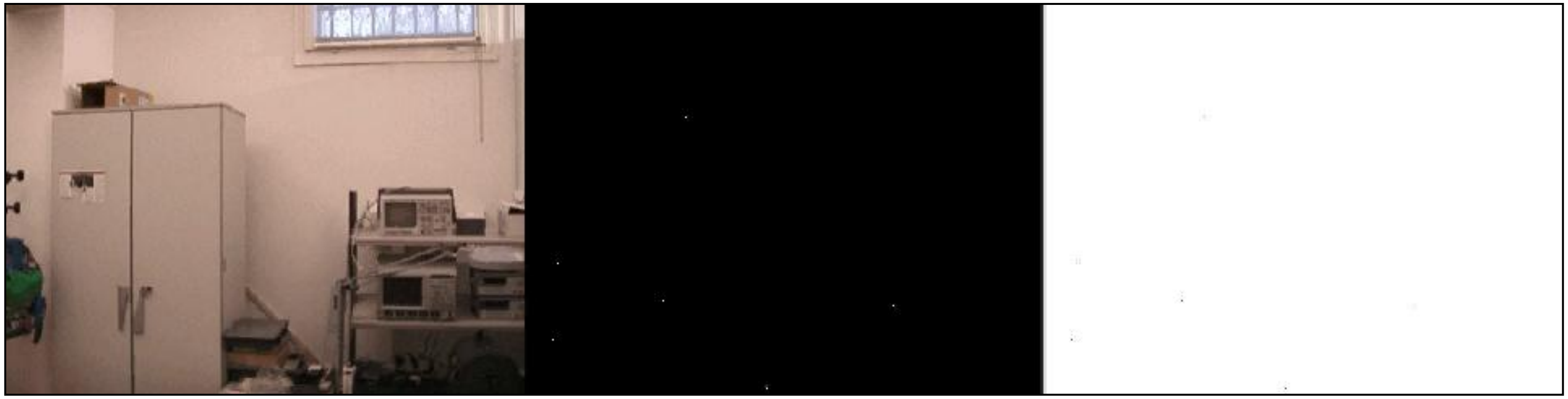
$$C_{t-1}(i,j) = \begin{cases} 255 & \text{if} \quad C^I_t(i,j) = 255 \quad \text{and} \quad C^I_{t-1}(i,j) = 255 \\ 0 & \text{otherwise} \end{cases}$$

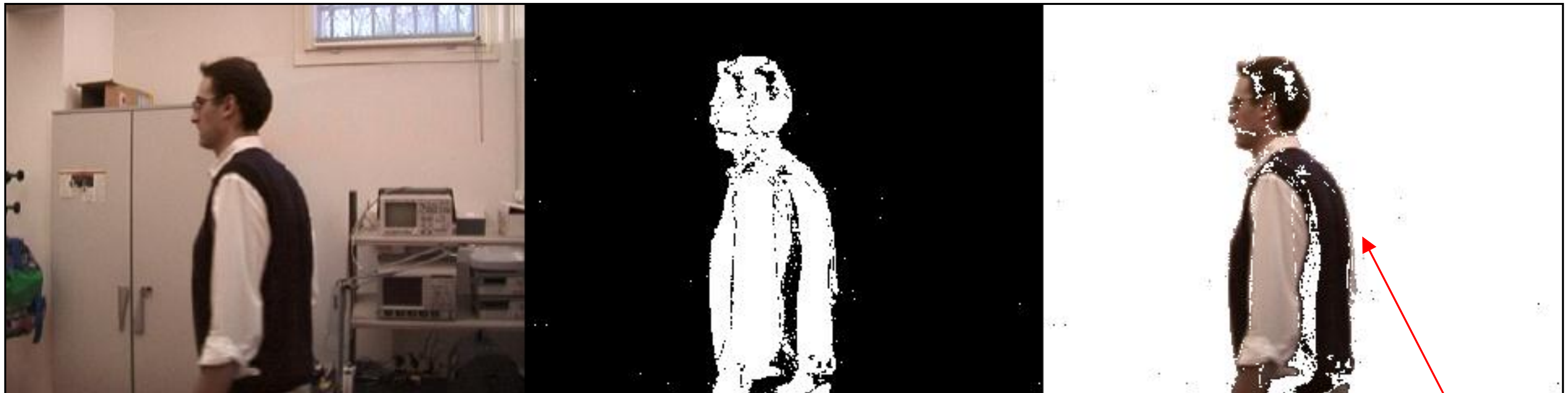


$$C_{t-1}(i,j)=\begin{cases}255 & \text{if} \quad C^I_t(i,j)=255 \quad \text{and} \quad C^I_{t-1}(i,j)=255 \\ 0 & \text{otherwise}\end{cases}$$

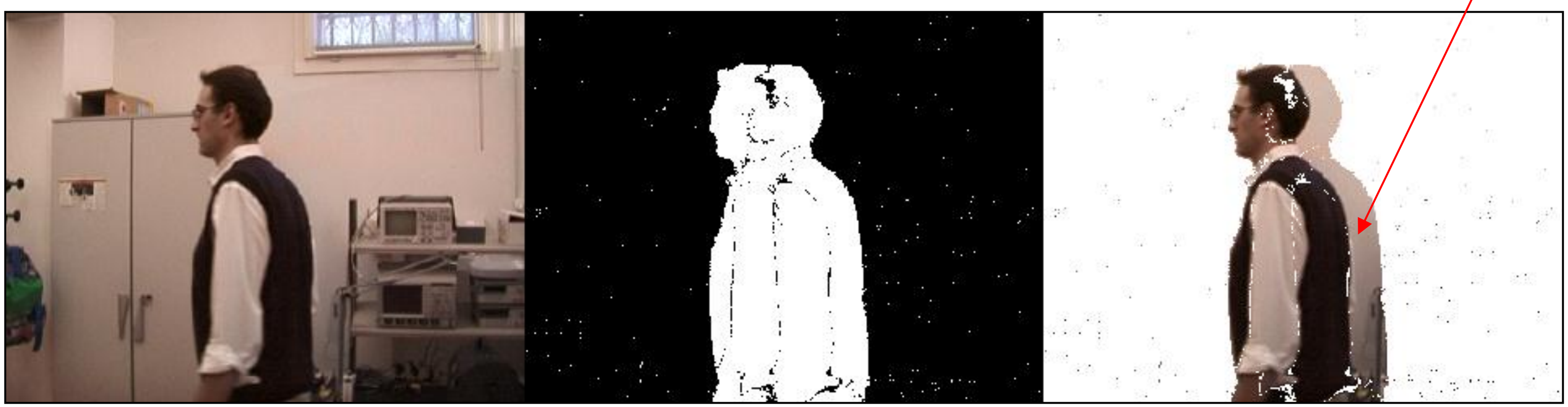


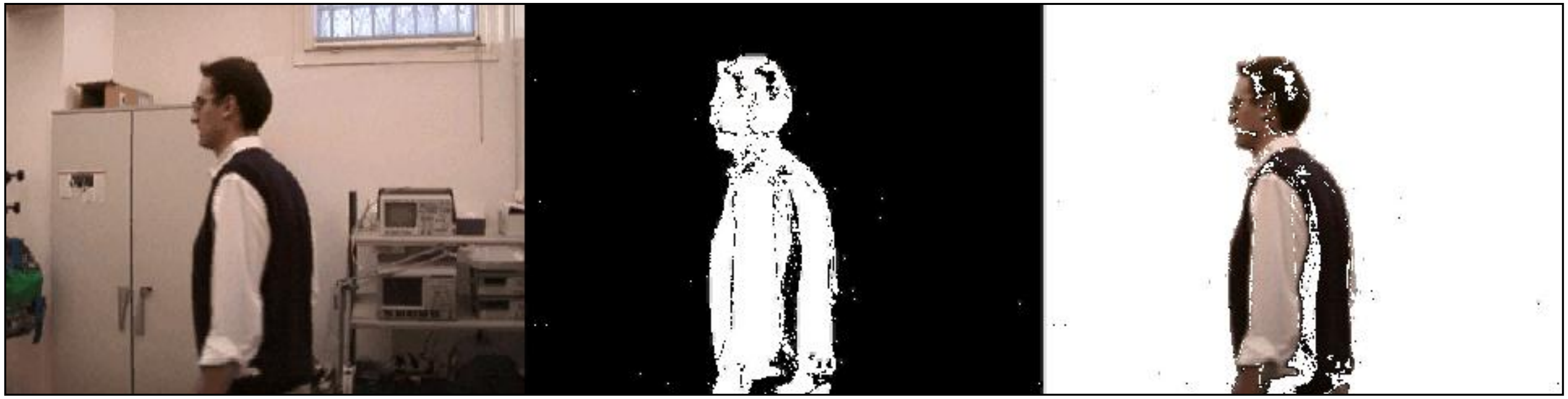






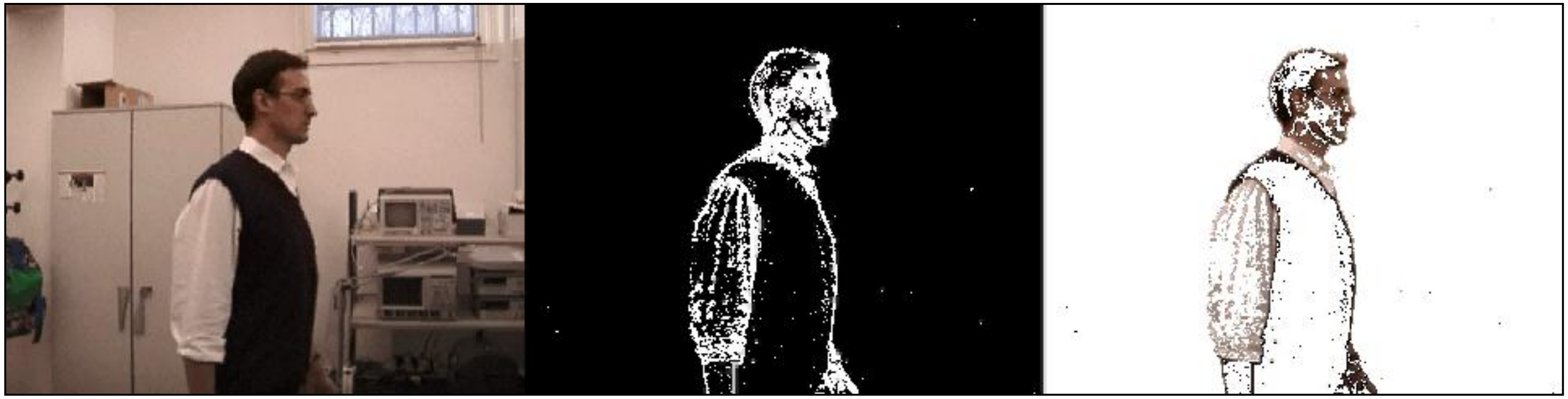
ghosting problem solved

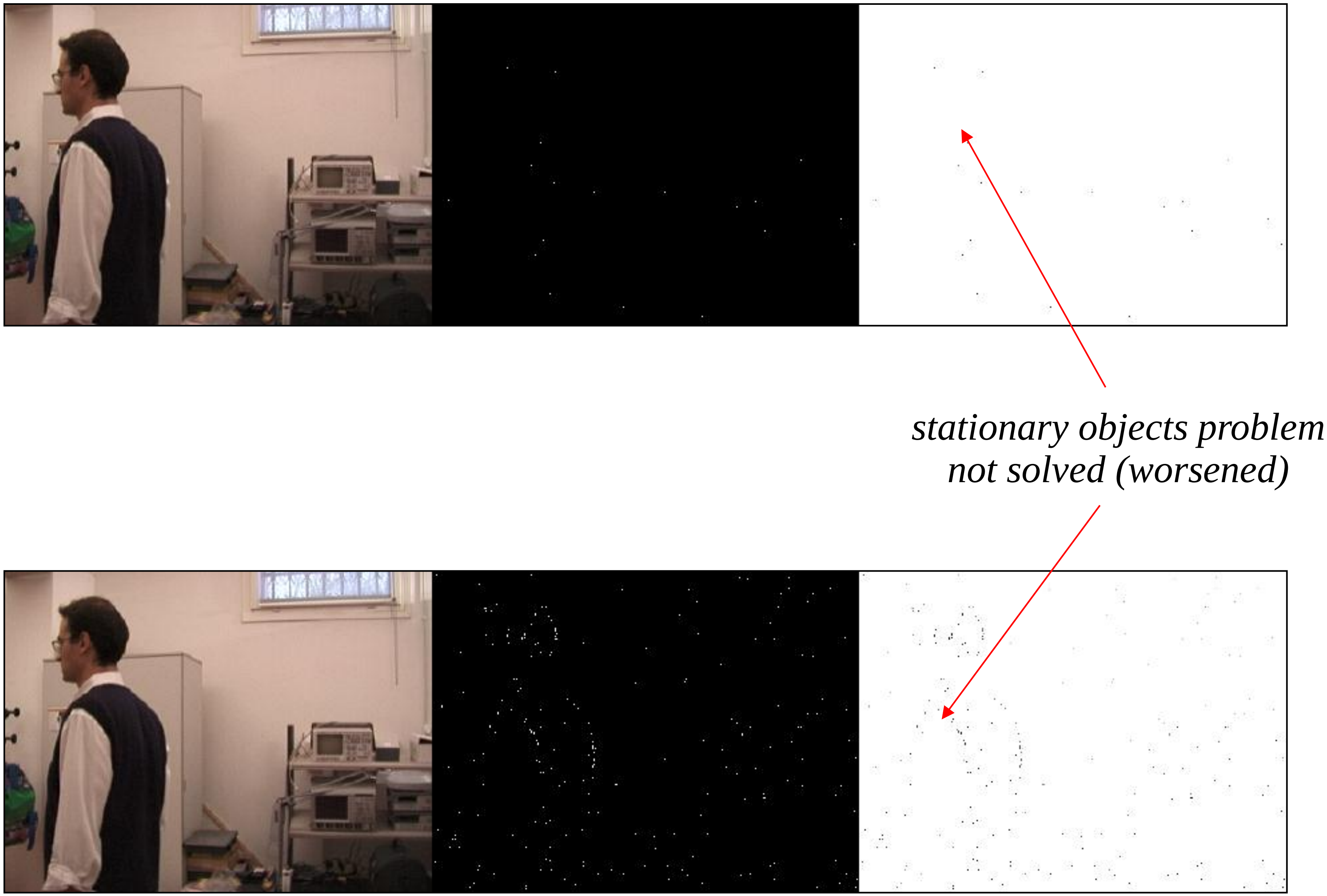




foreground aperture problem
not solved (worsened)

 - **Three-frame difference → results**




stationary objects problem
not solved (worsened)

• very similar to the two-frame difference: the current frame is compared with a “model” (e.g. an image) of the background of the monitored scene (instead of the previous frame):

$$C_t(i,j) = \begin{cases} 255 & if \quad D_t(i,j) = d\left(\vec{F}_t(i,j), \vec{B}_t(i,j)\right) > T \\ 0 & otherwise \end{cases}$$

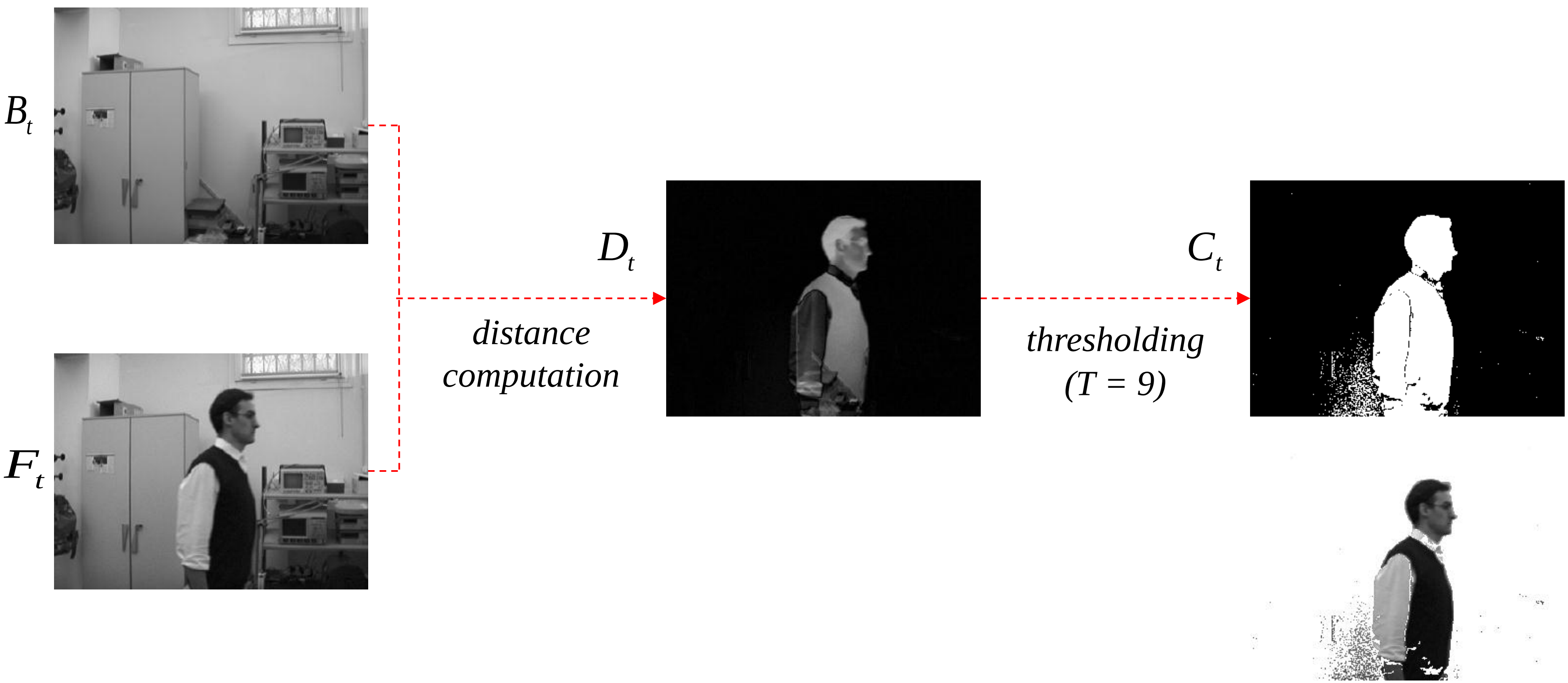
• in particular (distance based on the norm of the difference vector):

$$C_t(i,j) = \begin{cases} 255 & if \quad D_t(i,j) = \left\| \vec{F}_t(i,j) - \vec{B}_t(i,j) \right\|_p > T \\ 0 & otherwise \end{cases}$$

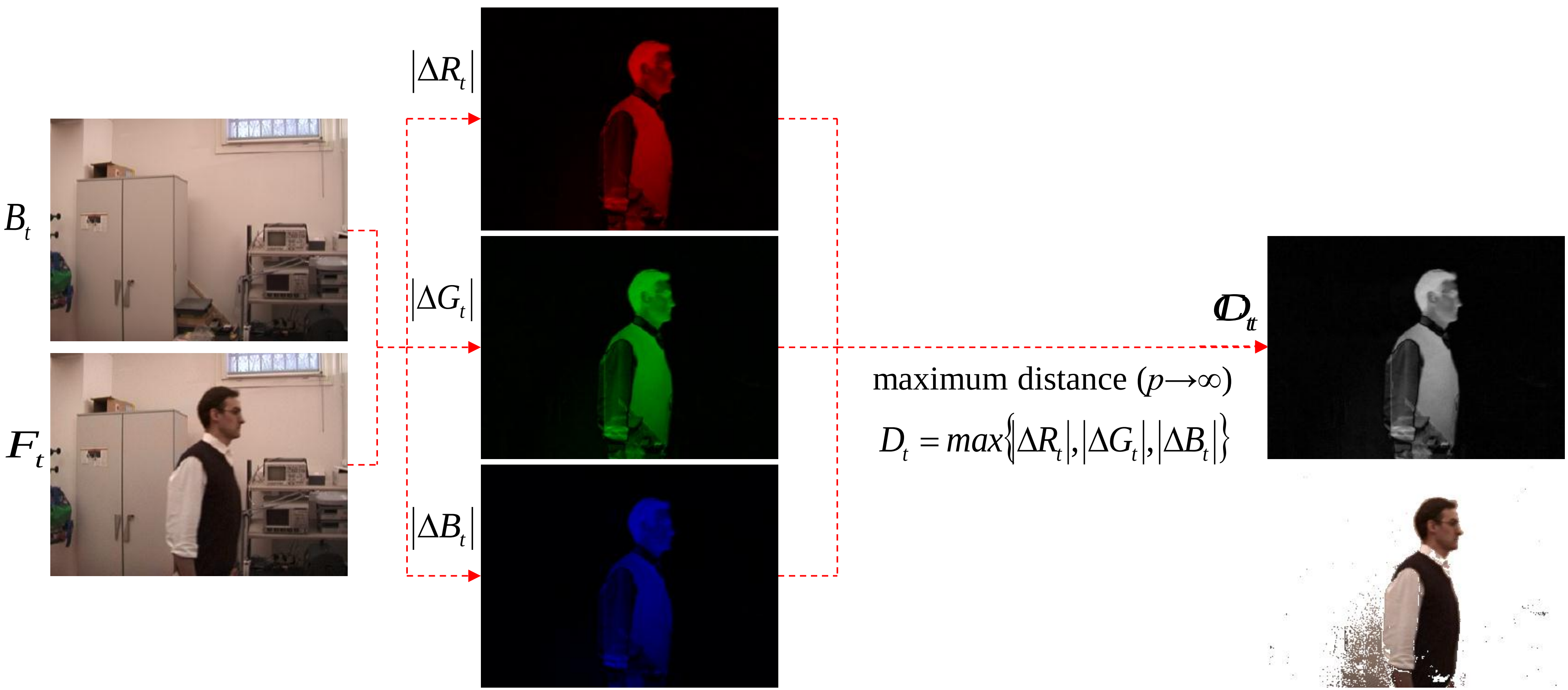
• where:

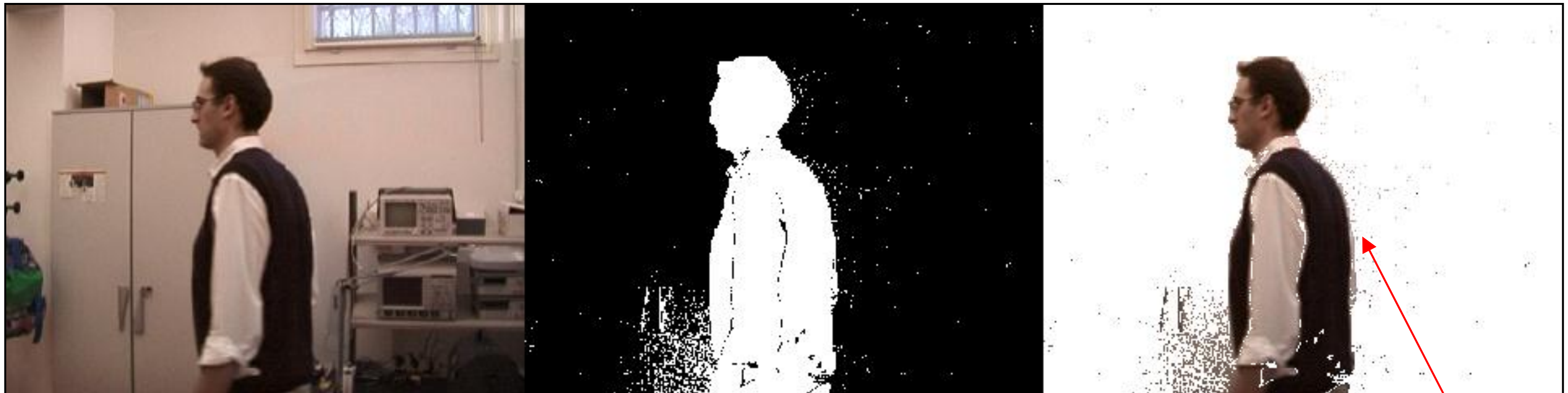
$$\vec{B}_t(i,j) = g\left(\vec{F}_{t-1}, \vec{F}_{t-2}, \ldots, \vec{F}_{t-k}\right)$$

$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = \left| F_t(i,j) - B_t(i,j) \right| > T \\ 0 & \text{otherwise} \end{cases}$$

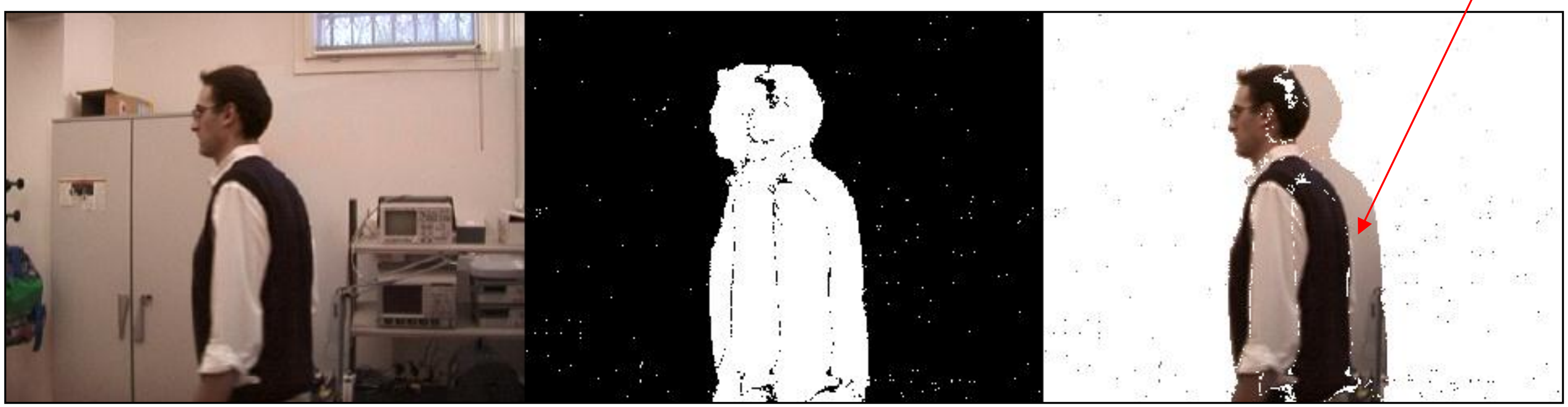


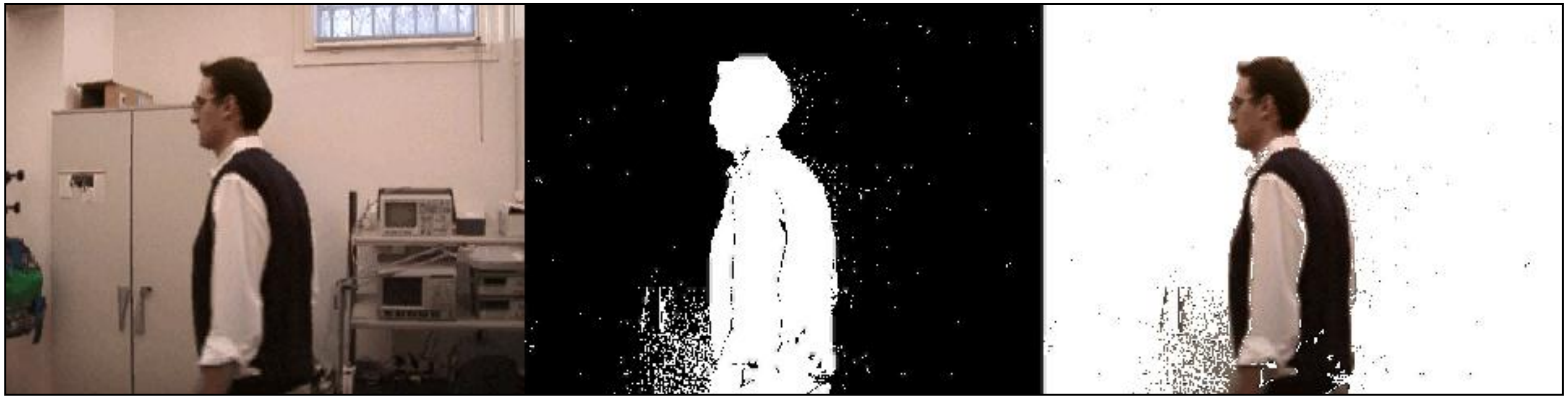
$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = \sqrt[p]{\left|\Delta R_t(i,j)\right|^p + \left|\Delta G_t(i,j)\right|^p + \left|\Delta B_t(i,j)\right|^p} > T \\ 0 & \text{otherwise} \end{cases}$$



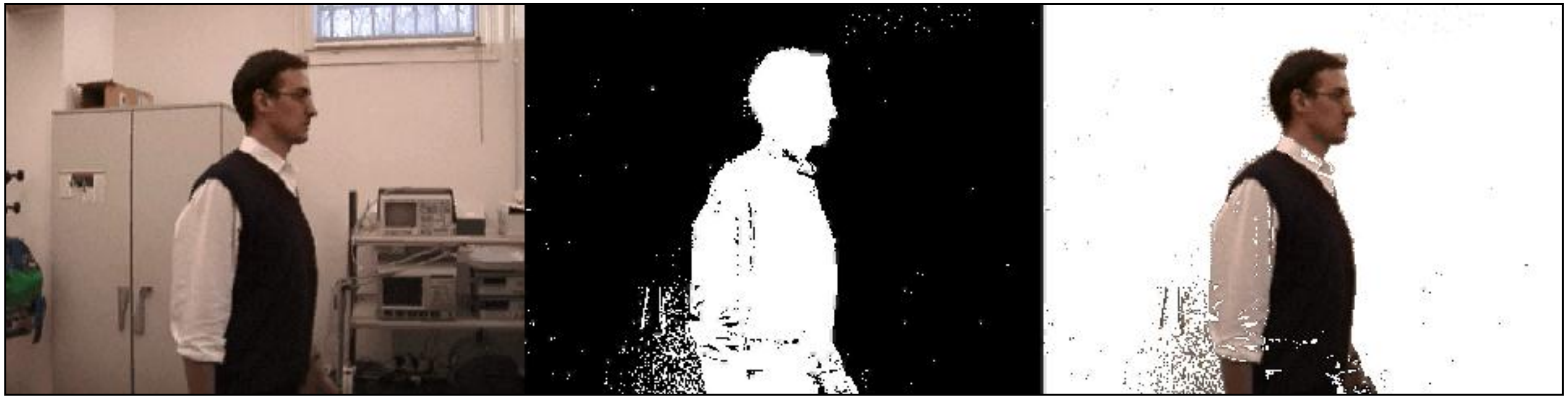


ghosting problem solved





foreground aperture problem solved



camouflage
(false negatives):
when the objects present (locally) a similar appearance (gray level or color) to the background

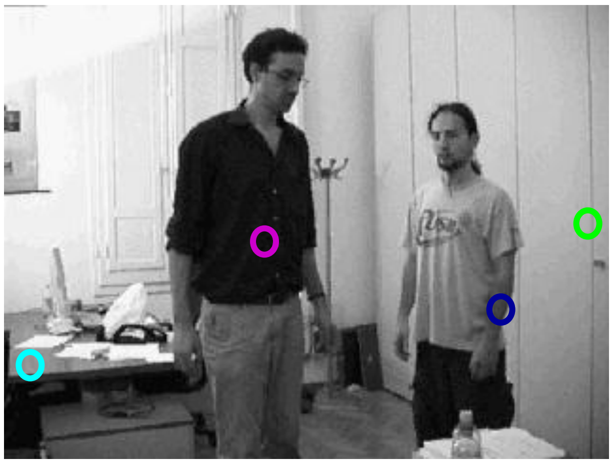
stationary objects problem solved

• sequence of 700 frames, sampled at 12,5 frames/s, 320x240 pixels, gray level and RGB:

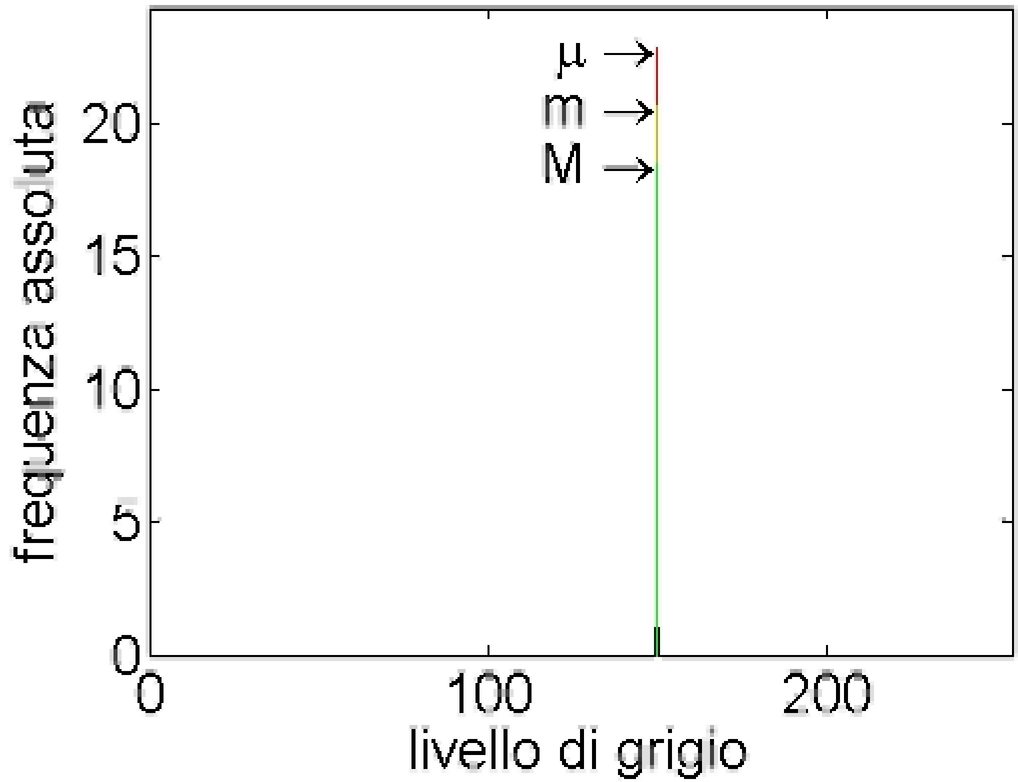
• Salient features:

- moving "objects" are poorly textured (→ foreground aperture);
- "objects" moving with a wide range of velocities (→ ghosting + disapp.); → temporal frame difference…bad results
- no initial subsequence free of moving "objects" → background initialisation
- non-stationary illumination conditions (gradual darkening) → background updating
- static background.

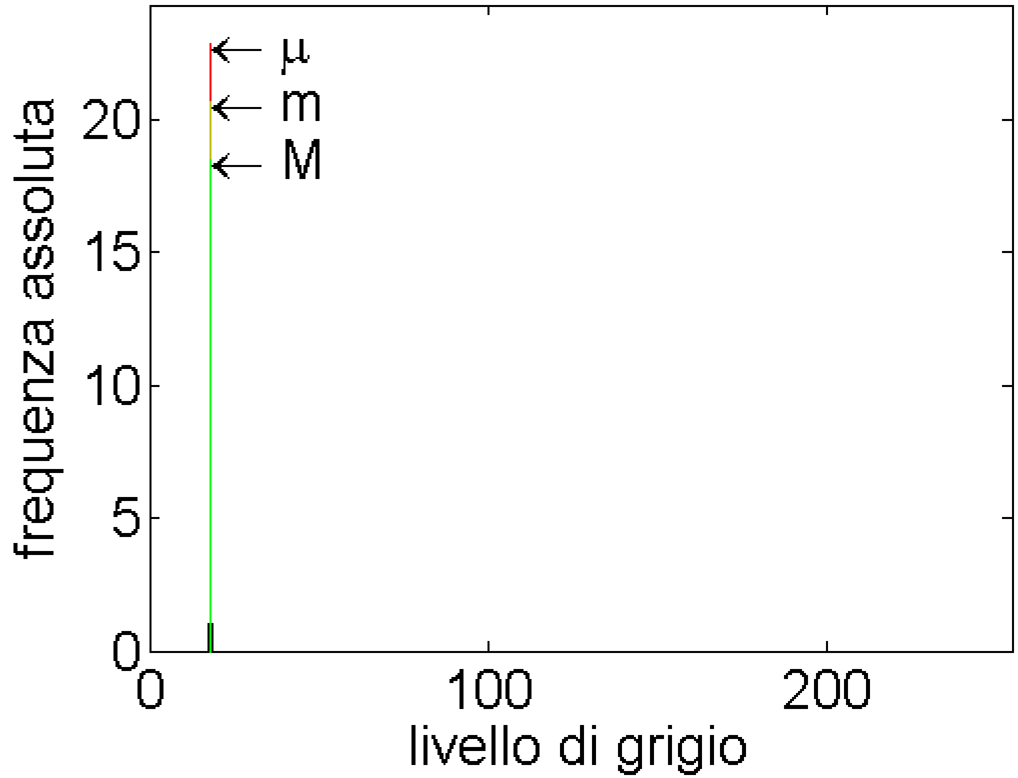
- we try to infer an image (in general, a model) of the background from the first 100 frames.
- idea: we build temporal statistics (histograms) of the intensities at each pixel, then we compute an estimate of the background...the estimate must be robust to possible foreground samples!



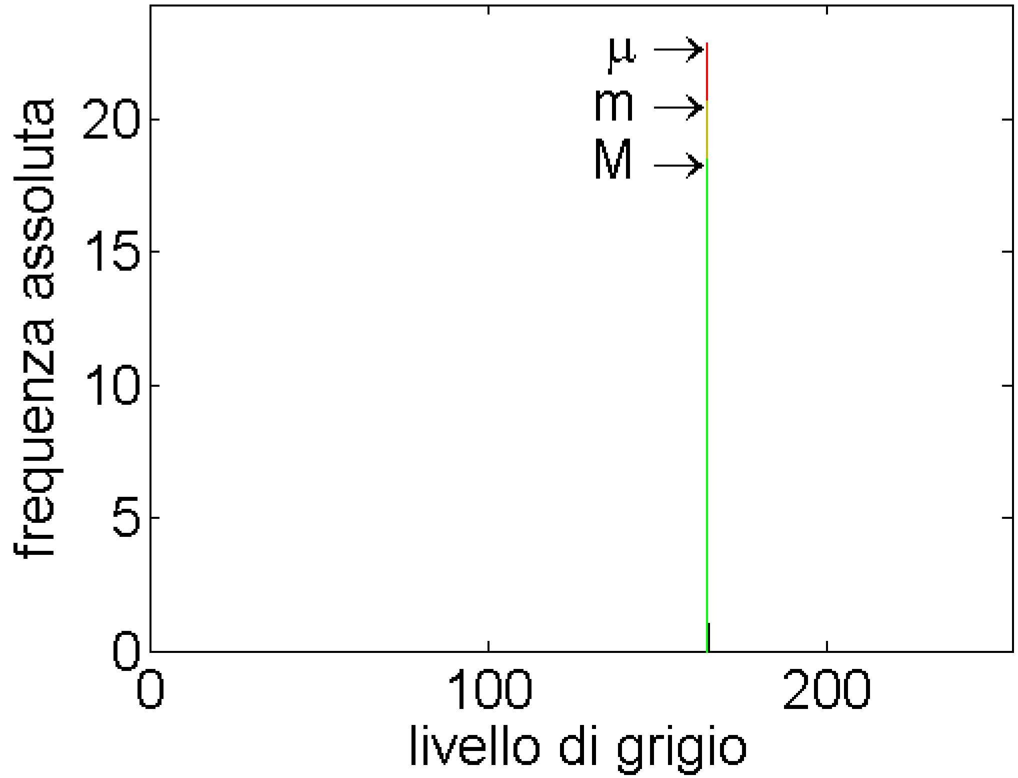
$\mu \rightarrow$ mean
m $\rightarrow$ mode
M $\rightarrow$ median

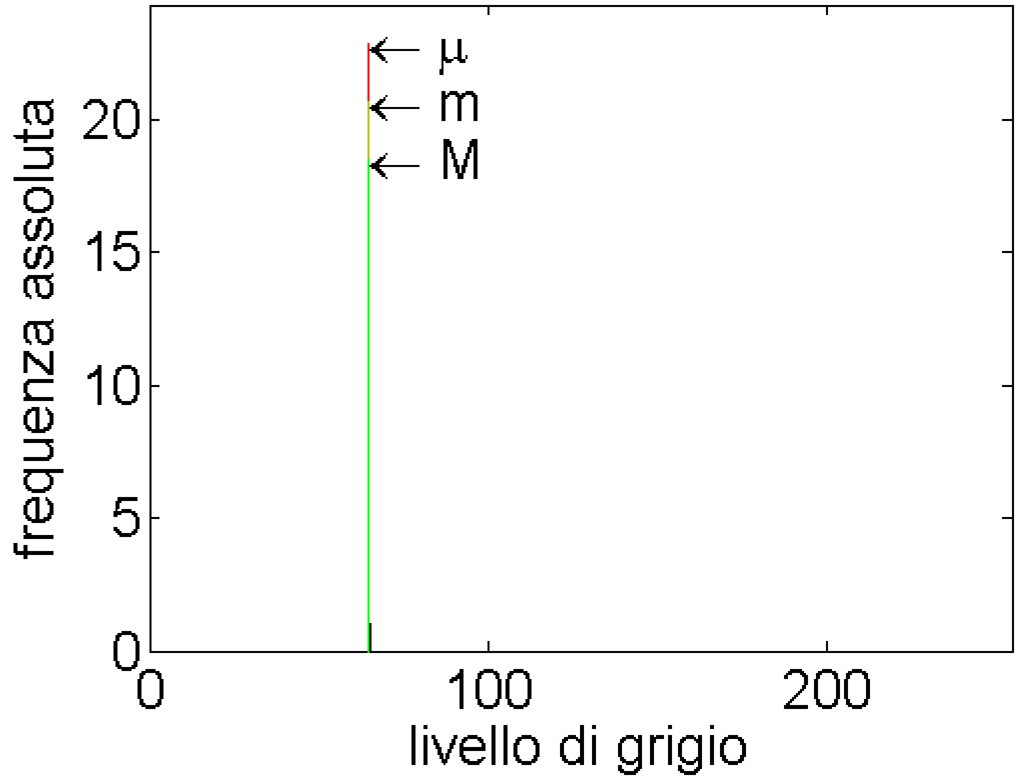


μ → mean
m → mode
M → median



- $\mu \rightarrow$ mean
- $m \rightarrow$ mode
- $M \rightarrow$ median





μ → mean
m → mode
M → median

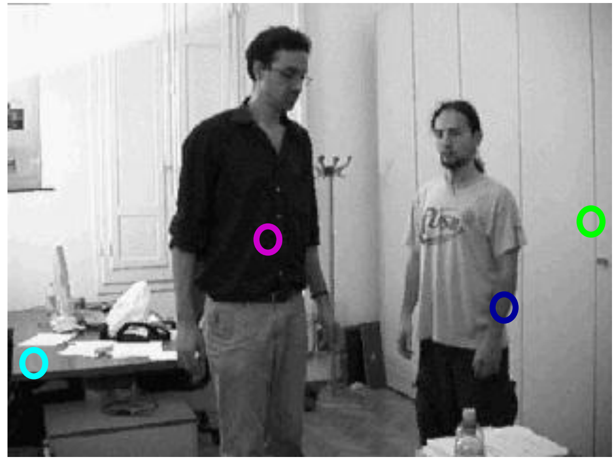
“blind” mean:

- "blind" initialization: at each pixel, the entire time series of sample intensities is “blindly” considered

“blind” mode:

“blind” median:

• "selective" initialization: at each pixel, only background samples, as classified by temporal frame difference + “conservative” morphology (e.g. dilation + filling), are “selectively” considered



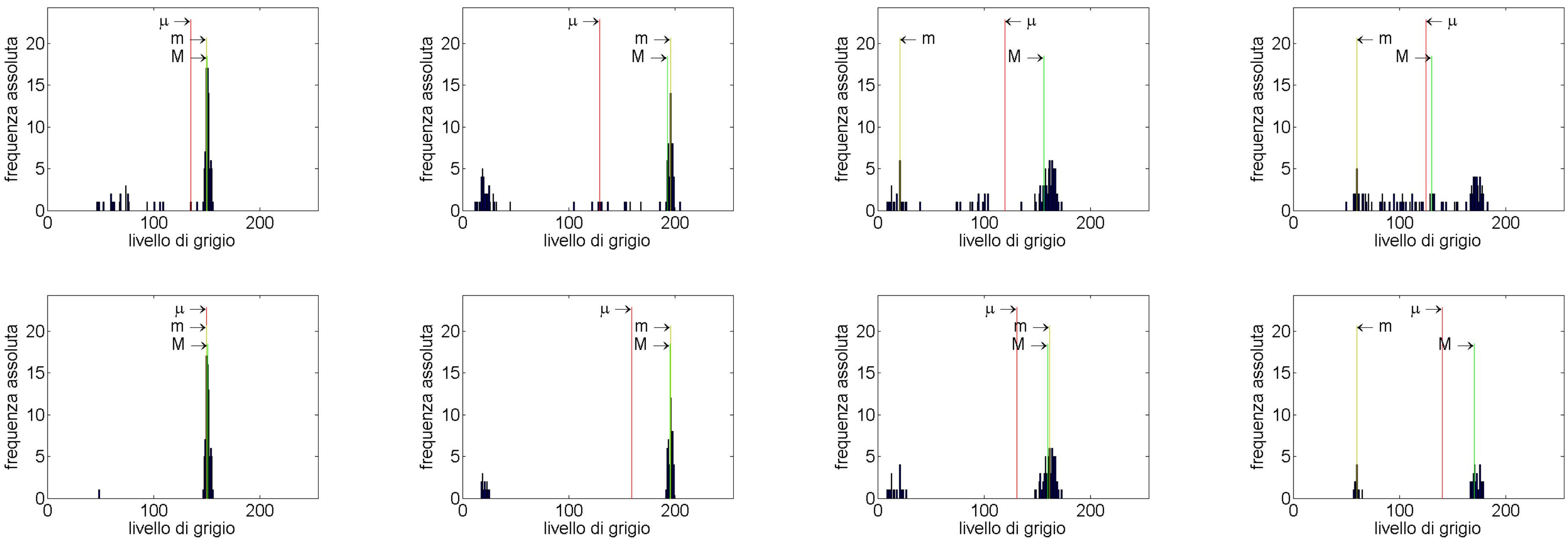
μ → mean

m → mode

M → median

blind

selective



“selective” mean:

“selective” mode:

“selective” median:

• the generated background can thus be used to perform background subtraction at next frames

• even a slight and gradual illumination change yields the "explosion" of change masks

• necessity to update the background (background updating)

• in general:

$$\vec{B}_{t+1}(i,j) = g\left(\vec{F}_t, \vec{F}_{t-1}, \ldots, \vec{F}_{t-k}\right)$$

it can be "very big"

• commonly used since they represent a very good tradeoff between effectiveness and computational efficiency...recursive procedures:

$$\vec{B}_{t+1}(i,j) = r\left(\vec{B}_t(i,j), \vec{F}_t\right)$$

• in particular, the so-called "alpha-blending" updating procedure:

• "blind":

$$\vec{B}_{t+1}(i,j) = \alpha \cdot \vec{F}_t(i,j) + (1-\alpha) \cdot \vec{B}_t(i,j) \quad \forall (i,j)$$

change mask: output of background subtraction + morphology

• "selective":

$$\vec{B}_{t+1}(i,j) = \begin{cases} \alpha \cdot \vec{F}_t(i,j) + (1-\alpha) \cdot \vec{B}_t(i,j) & if \quad C_t(i,j) = 0 \\ \vec{B}_t(i,j) & otherwise \end{cases}$$

•where $\alpha \in [0,1]$,called "adaptation rate", represents the speed of adaptation of the background model to changes occurring in the monitored scene.

• "blind" updating with $\alpha = 0.2$

• illumination changes are effectively "worked out"; however, since the background model "blindly" incorporates (blending) foreground samples, temporal frame-difference problems reappear under different guises (ghosting, foreground aperture and disappearance of stationary objects; in fact $\alpha = 1 \rightarrow$ background subtraction $\equiv$ two-frame difference)

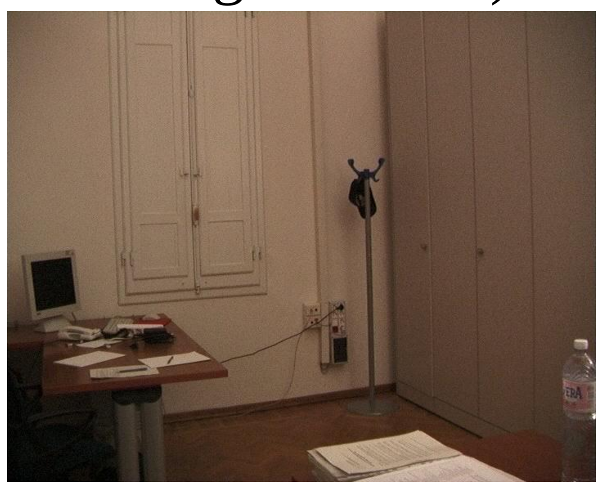
• necessity to update selectively

• "selective" updating with $\alpha = 0.2$

• illumination changes are effectively "worked out"; moreover, the problems of ghosting, foreground aperture and disappearance of stationary objects are solved.

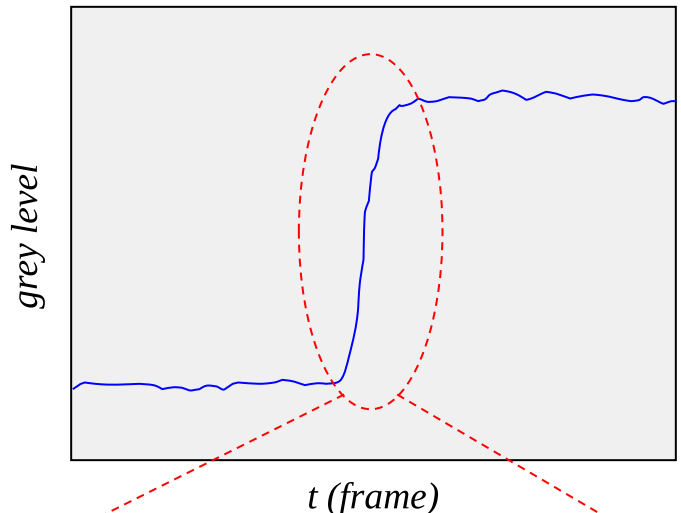
• multi-appearance background (waving trees, rain, snow, monitor flickering)

• sudden illumination changes (sun suddenly covered by clouds, switching lights on/off)



• When illumination changes are very fast (with respect to the frame capturing rate), previous methods fail since they rely on a pixel-wise time-adaptive modeling of background appearance.

pixel (i,j)



foreground object?

sudden illumination change?

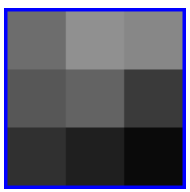
it is very difficult to discriminate !!!

- When illumination changes are very fast (with respect to the frame capturing rate), previous methods fail since they rely on a pixel-wise time-adaptive modeling of background appearance.
- Neighborhood-based approaches: pixels are classified as background/foreground by comparing the intensities within a neighborhood in the background (fixed) and in the current frame.

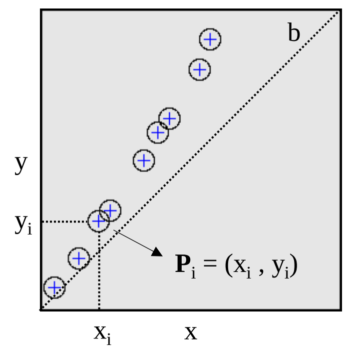
B

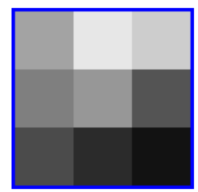
F_t

$\mathbf{X}$



x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9





y_1	y_2	y_3
y_4	y_5	y_6
y_7	y_8	y_9

$\mathbf{Y}_t(i, j)$

$$C_t(i,j) = \begin{cases} 255 & if \quad D_t(i,j) = d\left(\mathbf{X}, \mathbf{Y}_t(i,j)\right) > T \\ 0 & otherwise \end{cases}$$

- The effects of possible disturbance factors must be modeled and incorporated into the "distance" function *d(·, ·):* foreground pixels are thus detected "a-contrario"!

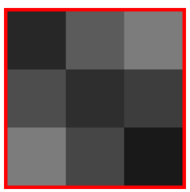
• When illumination changes are very fast (with respect to the frame capturing rate), previous methods fail since they rely on a pixel-wise time-adaptive modeling of background appearance.

• Neighborhood-based approaches: pixels are classified as background/foreground by comparing the intensities within a neighborhood in the background (fixed) and in the current frame.

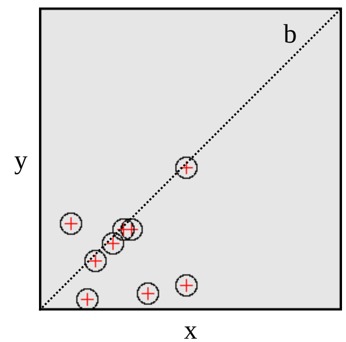
B

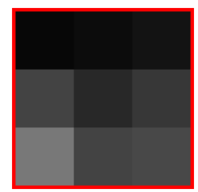
F_t

$\mathbf{X}$



x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9





y_1	y_2	y_3
y_4	y_5	y_6
y_7	y_8	y_9

$\mathbf{Y}_t(i, j)$

$$C_t(i, j) = \begin{cases} 255 & \textit{if} \quad D_t(i, j) = d(\mathbf{X}, \mathbf{Y}_t(i, j)) > T \\ 0 & \textit{otherwise} \end{cases}$$

• The effects of possible disturbance factors must be modeled and incorporated into the "distance" function *d(·, ·):* foreground pixels are thus detected "a-contrario"!

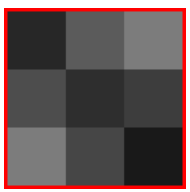
• When illumination changes are very fast (with respect to the frame capturing rate), previous methods fail since they rely on a pixel-wise time-adaptive modeling of background appearance.

• Neighborhood-based approaches: pixels are classified as background/foreground by comparing the intensities within a neighborhood in the background (fixed) and in the current frame.

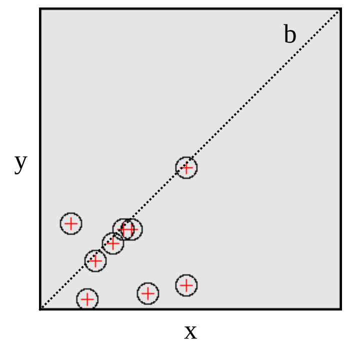
B

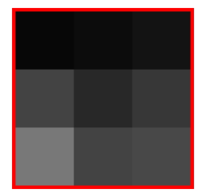
F_t

$\mathbf{X}$



x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9





y_1	y_2	y_3
y_4	y_5	y_6
y_7	y_8	y_9

$\mathbf{Y}_t(i, j)$

$$NCC: \quad C_t(i,j) = \begin{cases} 255 & \textit{if} \quad D_t(i,j) = \langle \mathbf{X}, \mathbf{Y}_t(i,j) \rangle \cdot \|\mathbf{X}\|^{-1} \cdot \|\mathbf{Y}_t(i,j)\|^{-1} > T \\ 0 & \textit{otherwise} \end{cases}$$

• The effects of possible disturbance factors must be modeled and incorporated into the "distance" function *d(·, ·):* foreground pixels are thus detected "a-contrario"!

THANK YOU !

pixel (i, j)

Old:

$$B_t(i,j) = b_{i,j,t} \in [0,255] \subset R$$

$$B_t(i,j) = f\left(F_{t-1}(i,j), F_{t-2}(i,j), \ldots, F_0(i,j)\right)$$

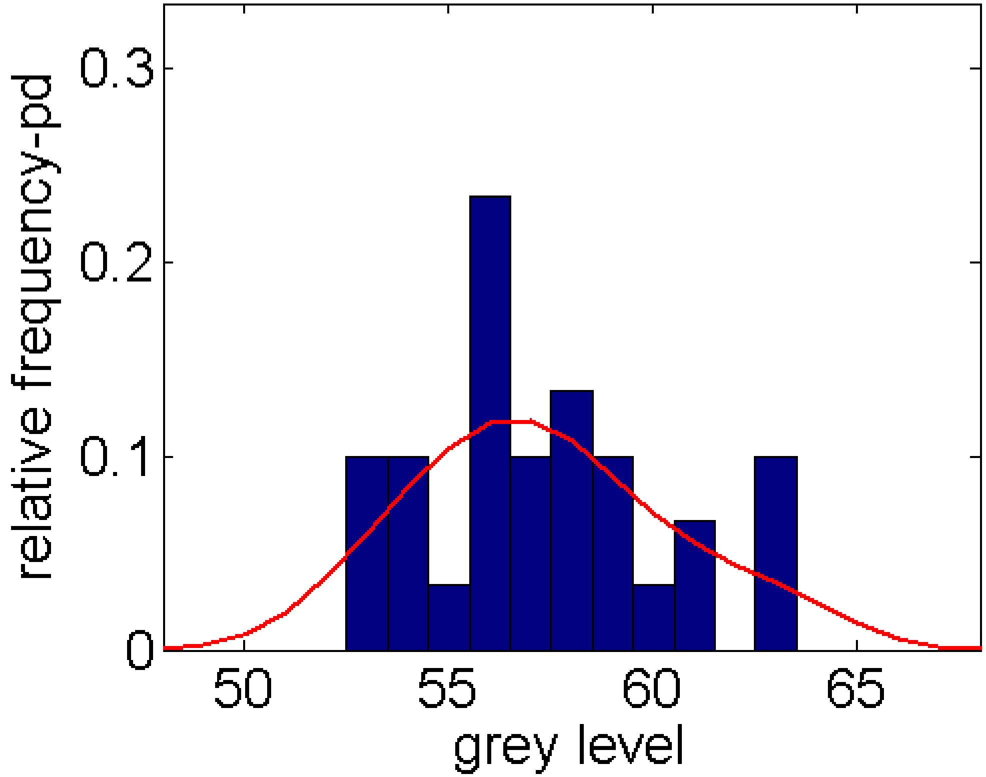
estimate of the background intensity value by weighted (exponentially) average of all past intensities

New:

$$B_t(i,j) = pdf_{i,j,t} : [0,255] \subset R \mapsto [0,+\infty]$$

$$B_t(i,j) = f\left(F_{t-1}(i,j), F_{t-2}(i,j), \ldots, F_{t-W}(i,j)\right)$$

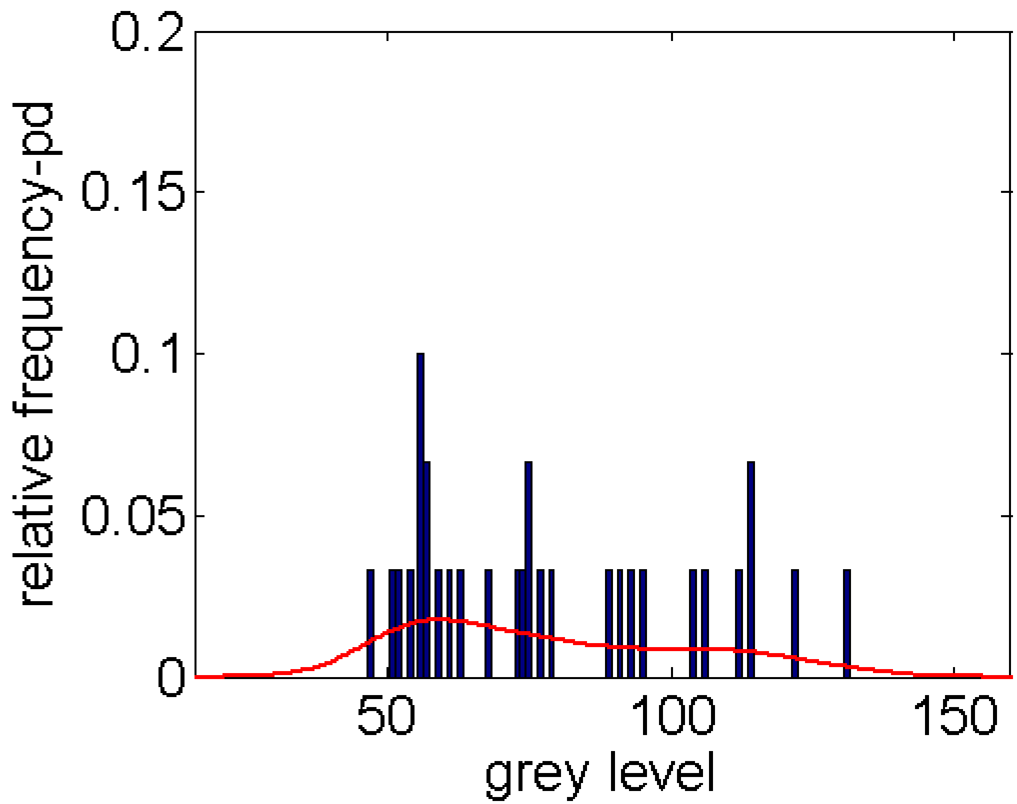
estimate of the background intensity pdf by (Gaussian) kernels applied to a finite window of W past intensities



$$pdf_{i,j,t} = f\left(F_{t-1}(i,j),\ldots,F_{t-30}(i,j)\right)$$

background / foreground classification:

$$C_t(i,j) = \begin{cases} 255 & \text{if} \quad D_t(i,j) = \left(1 - pdf_{i,j,t}\left(F_t(i,j)\right)\right) > T \\ 0 & \text{otherwise} \end{cases}$$



$$pdf_{i,j,t} = f\left(F_{t-1}(i,j),\ldots,F_{t-30}(i,j)\right)$$

FIFO update of the window of past intensities