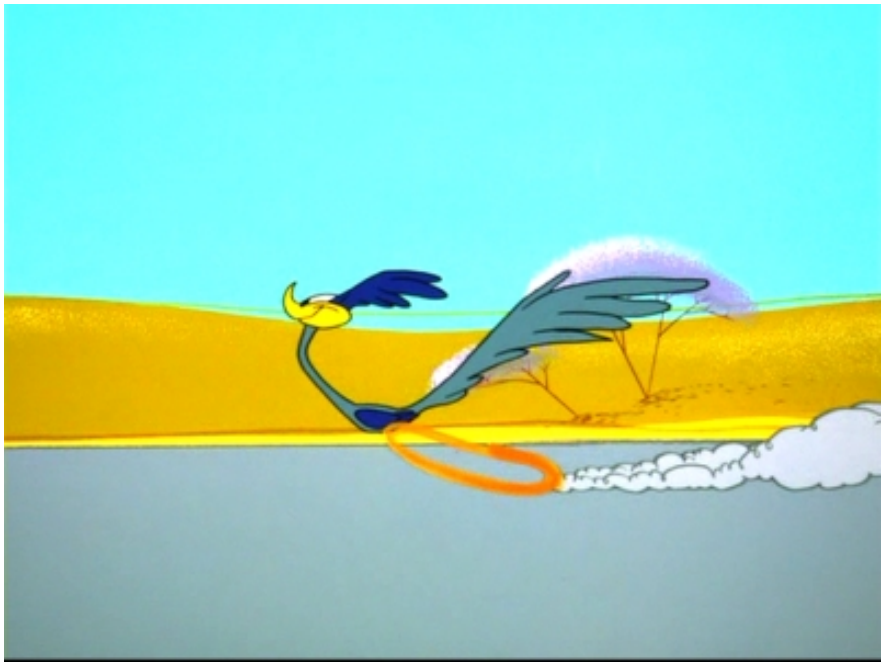
Università di Verona
Facoltà di Scienze MM.FF.NN.
Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

# Sistemi Avanzati per il Riconoscimento

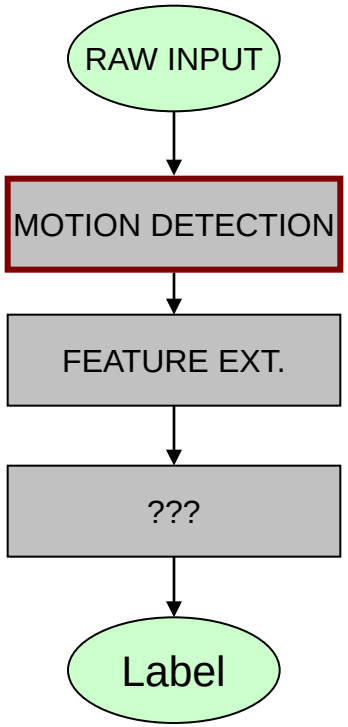


## Motion Detection

Dr. Marco Cristani

# Introduzione

- Motion Detection - operazione necessaria per la maggior parte delle applicazioni di sorveglianza, HCI avanzate, etc. che operano mediante movimento
  - Face-X
  - Tracking (face-hand-gesture-body)
- Tutte le applicazioni di cui sopra sono organizzate come architetture seriali
  - è importante non introdurre errori nei primi stage dell'architettura, pena la loro propagazione amplificata nei successivi
- Campo di ricerca attivo dal 1970, tuttora fervido
- Sinonimi di motion detection:
  - Sottrazione del background
  - Estrazione del foreground

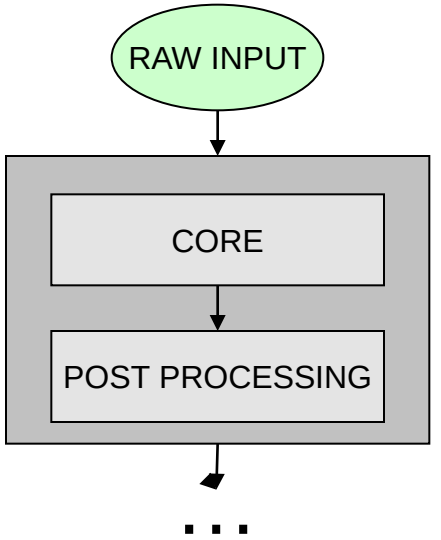


# Il problema

- Scopo (classico): Data una sequenza di frame acquisita da camera fissa, trovare tutti gli oggetti in movimento

- Meglio: data una scena acquisita da una telecamera mobile o fissa, evidenziare tutti gli oggetti inattesi (foreground, FG, visualizzato come pixel bianchi) distinguendoli da ciò che è atteso, la scena (background, BG, visualizzato come pixel neri)

- Motion detection di solito ha solo *post-processing*



# Come valutare le soluzioni

- Generalmente, è difficile avere un ground-truth (risultato corretto) per confrontare le soluzioni
- Non hanno molto senso esperimenti su dati sintetici
- Per valutare la bontà di un approccio di motion detection
  - falsi negativi *FN*
    - devono essere minimizzati se la perdita di un soggetto in movimento è grave
  - falsi positivi *FP*
    - devono essere minimizzati se l'identificazione di un oggetto in movimento corrisponde ad una scelta drastica da parte della macchina (lanciare un allarme nel caso della video-sorveglianza)
- Errore da valutare: E = FN + FP

# Soluzione intuitiva

- Identifico gli oggetti di foreground tramite la differenza tra il frame corrente $z^{(t)}$ e un'img $b^{(t)}$ rappresentante la scena

$$|z_i^{(t)} - b_i^{(t)}| > Th \quad , i \;\; pixel = 1,\ldots,I$$

- Primo problema: come ottenere l'immagine della scena in modo automatico
- Problema della **stima del background** (o background initialization)

# Soluzione intuitiva - difetto

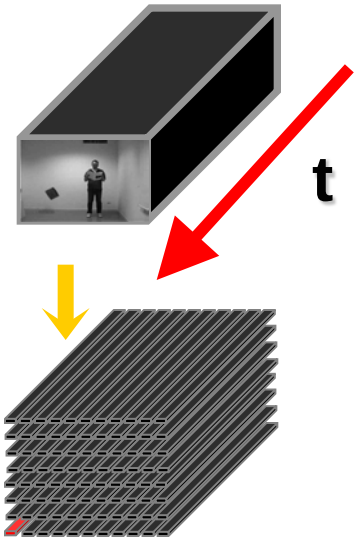
- L'immagine di background non è fissa ma si deve adattare a :
  - Cambi di illuminazione
    - graduali
    - repentini (lampi, nuvole etc.)
  - Movimenti
    - Oscillazioni della telecamera
    - Oggetti ad alta frequenza appartenenti alla scena (come rami di alberi,onde etc.)
  - Cambiamenti nella geometria della scena
    - Auto parcheggiate etc., ...
- Problema del **mantenimento del background** (o BG adaption)

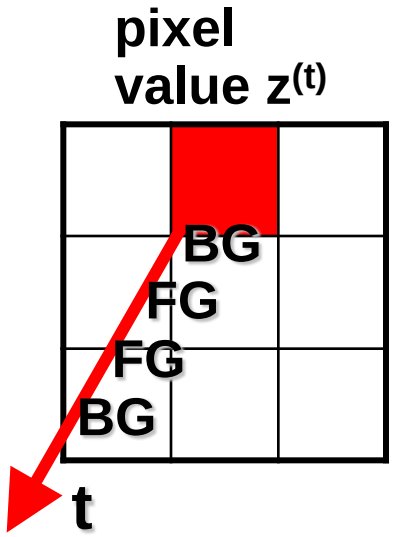
# Metodi di base

- Approcci
  - per pixel
  - per regione
  - per frame

# Metodi di base – approcci per-pixel

- Ogni pixel rappresenta un processo indipendente
- Ad ogni istante temporale t ogni pixel viene classificato come **FG** or **BG**



# Metodi di base

- Differenza tra frame:

$$|z_i^{(t)} - b_i^{(t)}| > Th$$

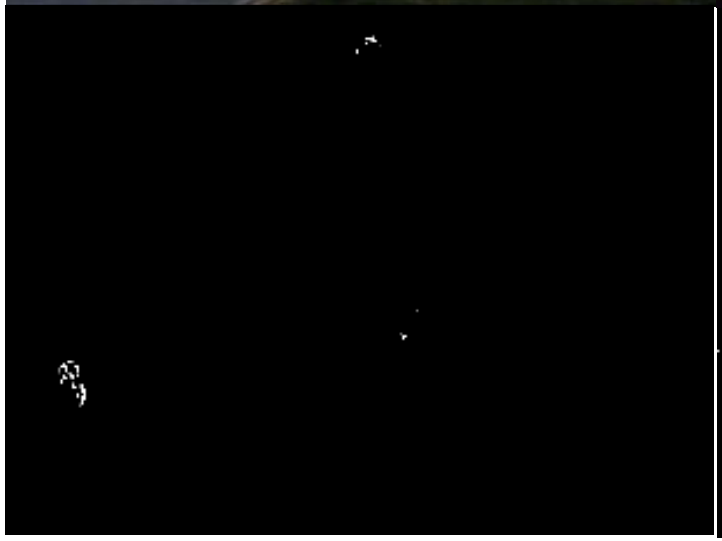
  - Il background stimato è semplicemente l'immagine precedente
  - Lavora solo in particolari condizioni (velocità degli oggetti in movimento e frame rate – fattori collegati)
  - Difficile trovare una soglia *Th* adatta (sensibilità)

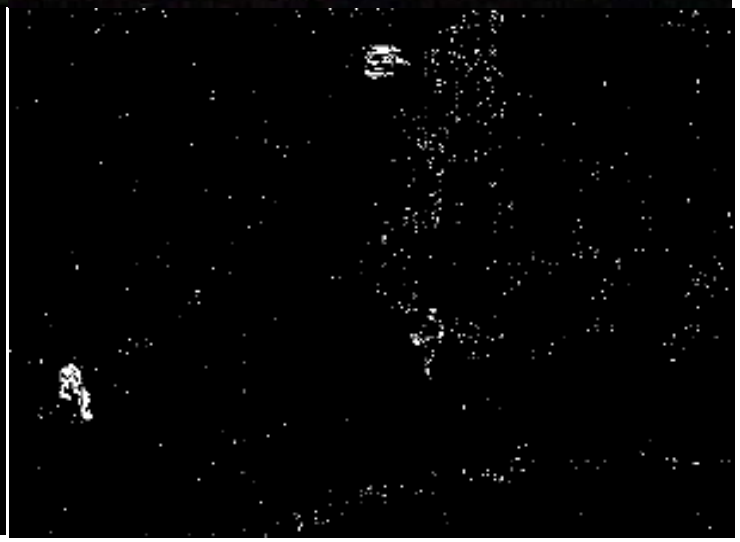
# Metodi di base

Sequenza

Diff. assoluta

Th: alta





Th: bassa

# Metodi di base: approccio basato su mediana

- Il background è la media o la mediana (Velastin, 2000; Cucchiara, 2003) degli n frame precedenti:
  - piuttosto veloce, ma esoso in termini di spazio: i requisiti di memoria sono n * size(frame)
- Modello il background come una *media mobile*:

$$b^{(t+1)}_i = \alpha * z^{(t+1)}_i + (1 - \alpha) * b^{(t)}_i$$

  - $\alpha$, il learning rate, è tipicamente basso - 0.05
  - nessun requisito di memoria

# Metodi di base: considerazioni

- Il modello di background ad ogni locazione di pixel è basato sulla storia recente del pixel stesso
- In molti lavori, tale storia è:
  - Semplicemente i primi n frame
  - Una media pesata in cui i valori più recenti hanno peso più grande
- In questi metodi non viene modellata alcuna correlazione tra pixel adiacenti (approcci per pixel)

# Metodi di base: selettività

- Ad ogni frame, un pixel viene classificato come BG o FG
- Che feedback puo' essere ricavato da questa operazione?
- →se il pixel viene identificato come foreground, esso non viene incluso nel modello di BG
- In questo modo il modello di BG non viene inquinato da valori che logicamente non appartengono alla scena
- Media mobile con selezione

$$b_i^{(t+1)} = \alpha z_i^{(t)} + (1-\alpha) B_i^{(t)} \quad \text{if } z_i^{(t)} \in \text{BG}$$

$$b_i^{(t+1)} = b_i^{(t)} \quad \text{if } z_i^{(t)} \in \text{FG}$$

- Accorgimento usato largamente
- Problemi
  - il BG modellato è monomodale
  - come scegliere la soglia Th

# Metodi di base: selezione della soglia

- Pfinder (Wren, Azarbayejani, Darrell, Pentland, 1997):
  - viene fittata una distribuzione gaussiana (μ,σ) sull'istogramma generato dalla storia di ogni pixel: questo restituisce la distribuzione (PDF) del background
  - aggiornamento dei parametri: media mobile

$$\mu_i^{(t+1)} = \alpha z_i^{(t)} + (1-\alpha)\mu_i^{(t)}$$

$$\sigma_i^{2\,(t+1)} = \alpha(z_i^{(t)} - \mu_i^{(t)})^2 + (1-\alpha)\sigma_i^{2\,(t)}$$

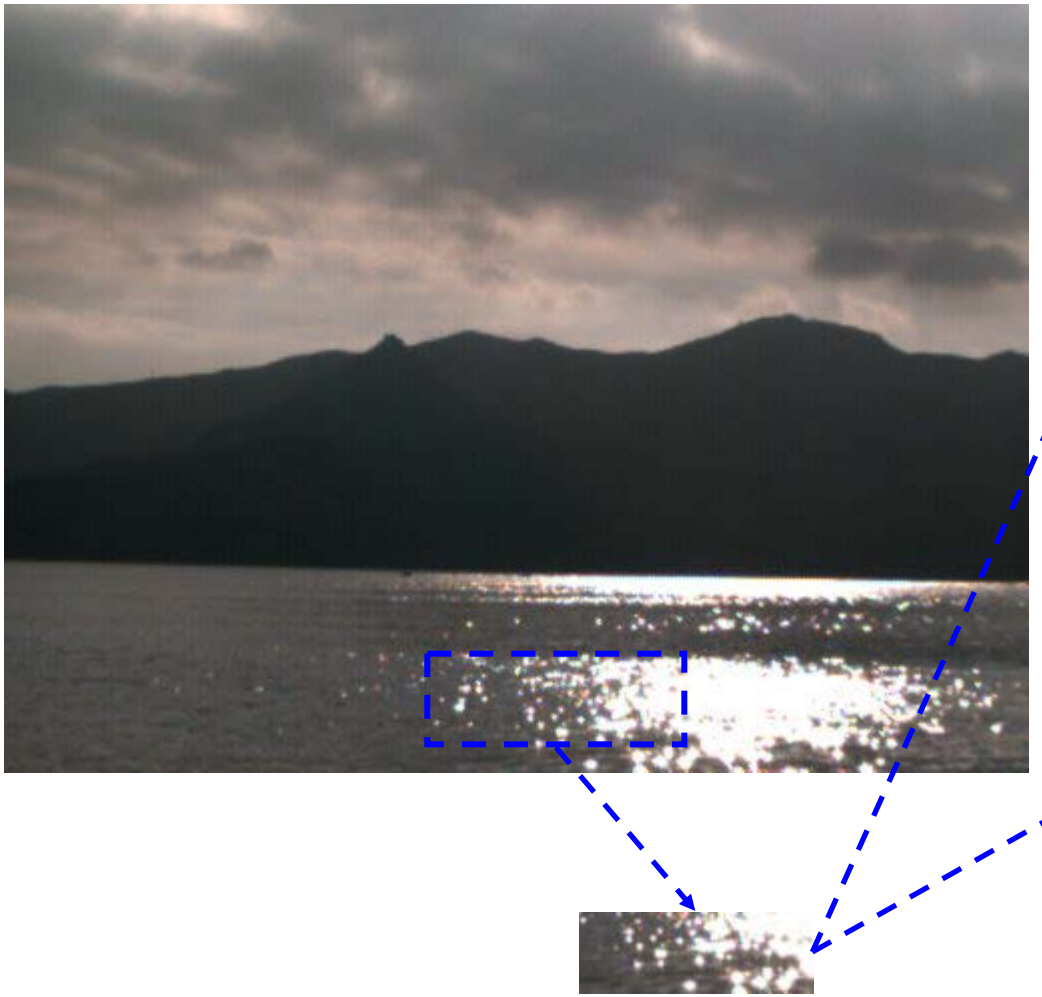
  - ciò risolve il problema della soglia: il *Foreground test* risulta

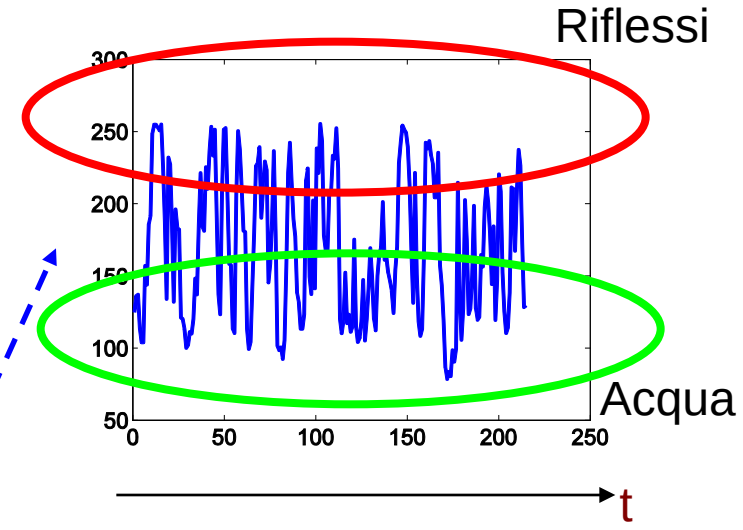
$$z_i^{(t)} \in FG \qquad \text{if } \left|z_i^{(t)} - \mu_i^{(t)}\right| > 2.5\sigma_i^{(t)} \textbf{= Th}$$

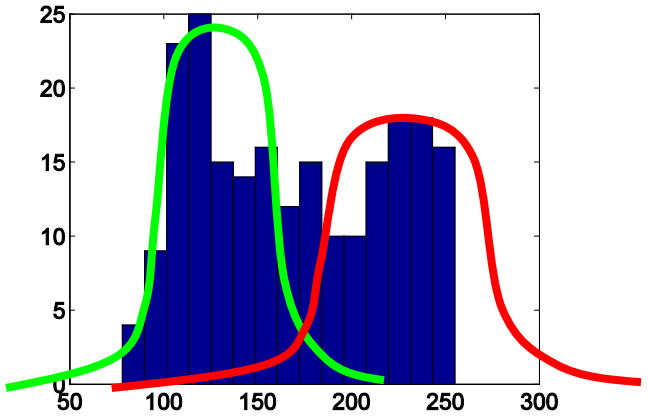
$$z_i^{(t)} \in BG \qquad \text{altrimenti}$$

NO MULTIMODALITA'

# Background bimodale





Stauffer, C., Grimson, W., 1999. Adaptive background mixture models for real-time tracking. In: Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition, Ft. Collins, CO, USA. pp. 246-252.

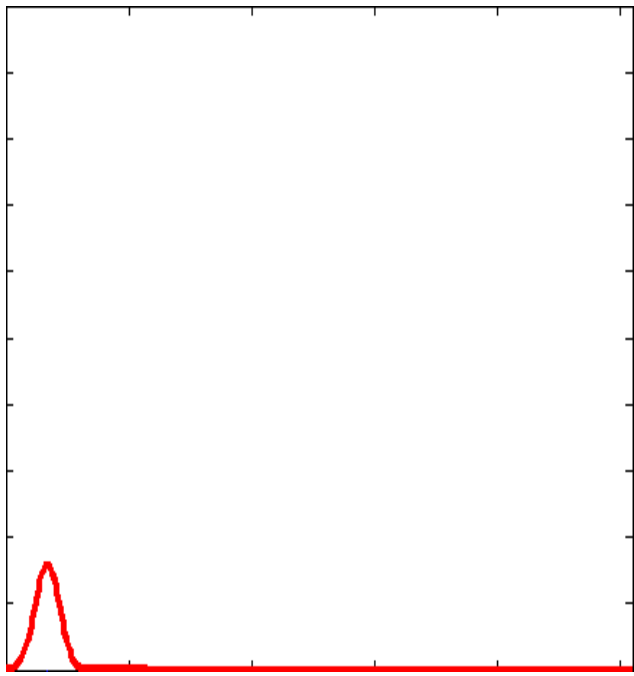
# TAPPMOG

- Modello statistico *generativo*: Time Adaptive Mixture Of Gaussian (TAPPMOG, Stauffer and Grimson, 1999)
- la probabilità di osservare il valore di pixel $z_i^{(t)}$ è

$$P(z_i^{(t)}) = \sum_{k=1}^{M} w_{ik}^{(t)} N(z_i^{(t)}; \mu_{ik}^{(t)}, \Sigma_{ik}^{(t)})$$

dove

  - $(\mu^{(t)}_{ik}, \Sigma^{(t)}_{ik})$ identificano le componenti gaussiane
  - $w^{(t)}_{ik}$ = *grado di importanza*; se alto, indica una componente che modella un segnale persistente, quindi di BG; altrimenti abbiamo un valore di FG;



# Misture di gaussiane

- Ad ogni nuovo frame, alcune Gaussiane associate al pixel i-esimo $z_i$ possono “matchare” il valore del pixel corrente, ossia

$$\left| z_i^{(t)} - \mu_{ik}^{(t)} \right| < 2.5\sigma_{ik}^{(t)}$$

  con k indice di componente
- considero come *matching component* la componente a distanza minore
- Tale gaussiana (e il rispettivo valore di pixel) viene classificata come BG o FG, mediante il *FG test* (ignoro l’indice i del pixel)

$$FG \quad \text{if} \quad \sum_{k=1}^{\text{matched}} w_k^{(t)} > Th$$

$$BG \quad \text{otherwise}$$

- Quindi $B = \underset{b}{\operatorname{argmin}} \sum_{k=1}^{b} w_k^{(t)} > Th$ è il numero di comp che modellano il BG

# Misture di gaussiane

- Nota importante: le componenti Gaussiane sono ordinate in maniera decrescente su $w_k^{(t)} / \sigma_k^{(t)}$

- Componenti con peso maggiore (viste più volte) con varianza minore (più stabili e regolari nel tempo) si trovano nelle prime posizioni

# Misture di gaussiane

- I parametri vengono addestrati on-line – il training avviene interfogliato con la classificazione
- I pesi di tutte le componenti di tutti i pixel vengono aggiornati come segue

$$w_k^{(t)} = (1-\alpha) w_k^{(t-1)} + \alpha M_k^{(t)}$$

$$\begin{cases} M_k^{(t)} = 1 \text{ if k è "matching comp."} \\ M_k^{(t)} = 0 \quad \text{altrimenti} \end{cases}$$

- Per un dato pixel, la *matching component* aumenta di peso, le altre diminuiscono

# Misture di gaussiane

- I parametri della matching component, $\mu_k$,$\sigma_k$, vengono aggiornate mediante media mobile, con le seguenti eq. di rinnovo:

$$\mu_k^{(t)} = (1-\alpha)\mu_k^{(t-1)} + \rho z^{(t)}$$

$$\sigma_k^{2(t)} = (1-\rho)\sigma_k^{2(t-1)} + \rho\left(z^{(t)} - \mu_k^{(t)}\right)^{\mathrm{T}}\left(z^{(t)} - \mu_k^{(t)}\right)$$

$$\rho = \alpha N\left(z^{(t)} \mid \mu^{(t)}, \sigma^{2(t)}\right)$$

- Se per un dato pixel non ho componenti che matchano, inserisco una nuova componente con

$$\mu_k^{(t)} = z^{(t)}$$

$$\sigma_k^{2(t)} = \text{valore alto}$$

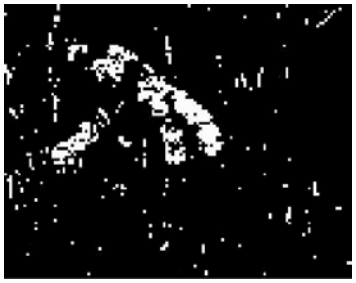
# Considerazioni importanti

- Il parametro $\alpha$ è detto learning rate, $\in [0,1]$
  - $\alpha$ alto:
    - PRO
      - recupero veloce degli errori
      - un oggetto inizialmente mobile, poi statico, viene “assorbito” presto dal modello di BG
    - CONTRO
      - comparsa di buchi in un oggetto mobile lento
  - $\alpha$ basso:
    - PRO
      - alta sensibilità verso oggetti lenti
    - CONTRO
      - poca adattività (difficoltà con cambi di illuminazione)

# Considerazioni importanti

- il numero di mode è definito una volta per tutte a priori (di solito da 3 a 5)
  - esistono metodi di scelta on-line per le componenti [Zivkovic@ICPR04]
- come inizializzare le Gaussiane ($\mu_{ik}$ ,$\Sigma_{ik}$)?

# Risultati (difficili)

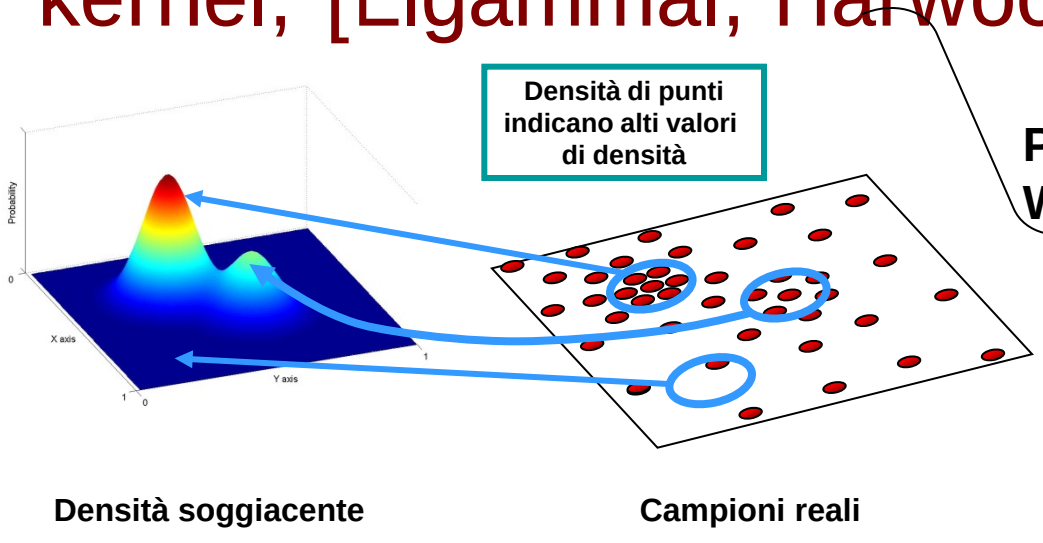


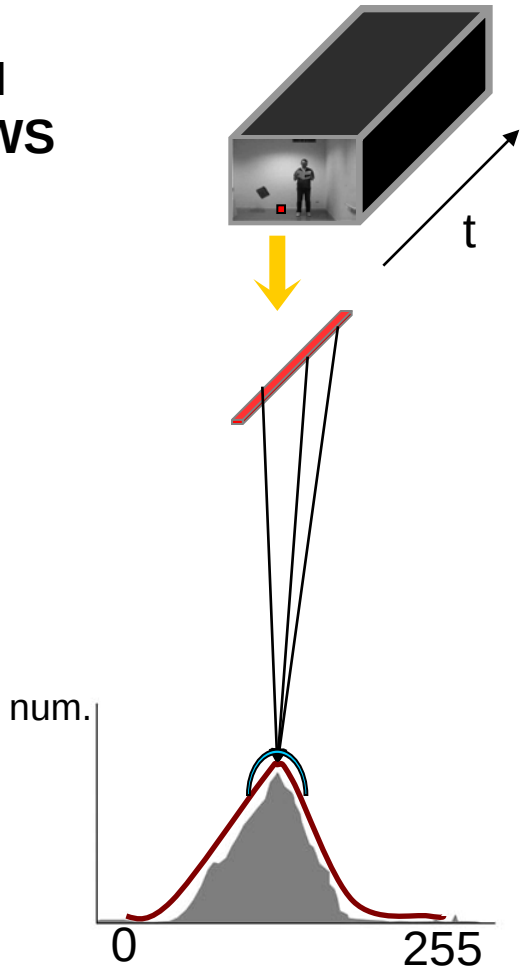
rumore!!!

$\alpha$ (troppo) basso

A. Elgammal, D. Harwood, L. S. Davis, "*Non-parametric Model for Background Subtraction*" IEEE ICCV99 Frame Rate Workshop. IEEE 7th International Conference on Computer Vision. Kerkyra, Greece, September 1999

# Stimatori non parametrici di BG (stimatori kernel, [Elgammal, Harwood, Davis, 99])




- La distribuzione sui valori di BG per un pixel è data dall'istogramma degli *n* valori di pixel più recenti
- Ogni valore della pdf
  - viene valutato puntualmente (stima *Parzen Windows*)
  - viene filtrato con una funzione particolare (*kernel*) in modo che la pdf risultante sia smooth

# Stimatori non parametrici di BG (stimatori kernel, [Elgammal, Harwood, Davis, 2000])

$$p\left(z^{(t)}\right) = \frac{1}{n}\sum_{i=1}^{n} K\left(z^{(t)} - z_i^{(t)}\right)$$

- K funzione kernel
- If $p(z^{(t)}) > Th$, il pixel z è classificato come BG
  - questo perchè il valore del pixel nel tempo non è cambiato “di molto”
- Gestisce multimodalità, e selettività
- Problema: i requisiti di memoria sono pesanti, ossia

(n * size(frame))* *il tempo di calcolo della probabilità*

# Limiti degli approcci per-pixel

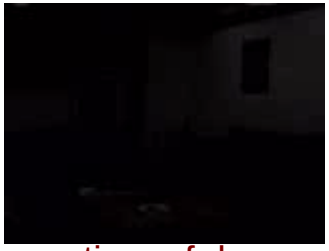
- Benchmark
  - Set di sequenze, ognuna delle quali contiene un problema tipico
    - Wallflower sequences


bootstrap


camouflage


foreground

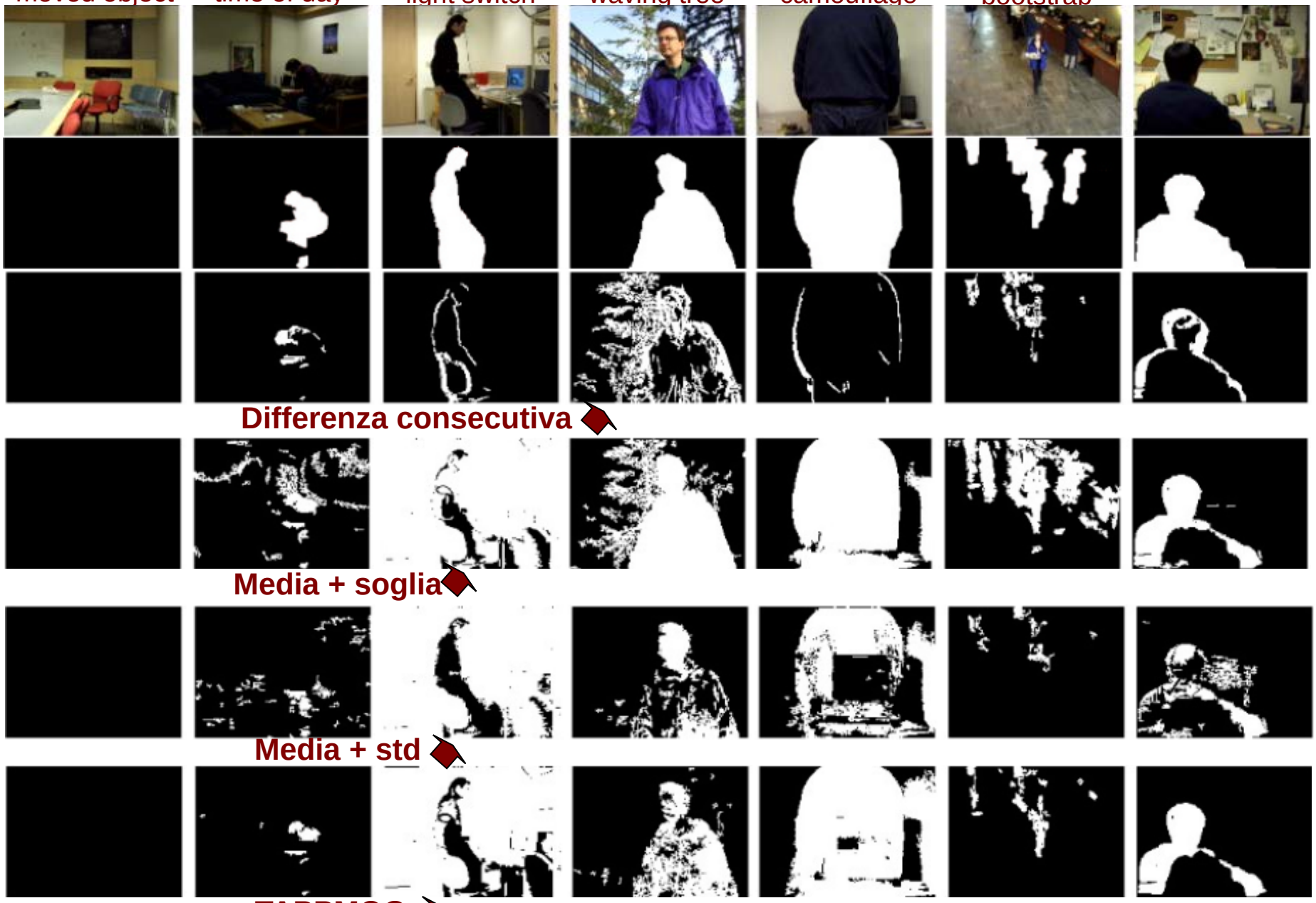

light switch


time of day


moved object

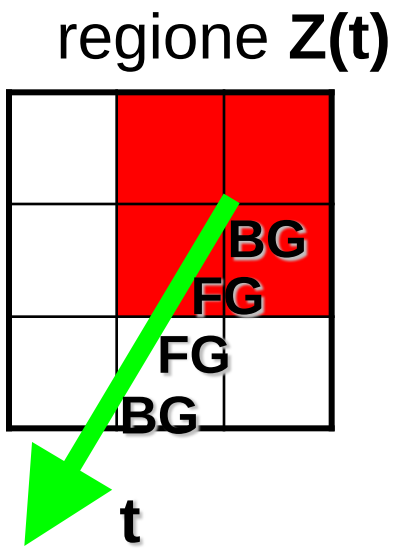

waving tree

**Kentaro Toyama, John Krumm, Barry Brumitt, Brian Meyers, "Wallflower: Principles and Practice of Background Maintenance", *Seventh International Conference on Computer Vision*, September 1999, Kerkyra, Greece, pp. 255-261, IEEE Computer Society Press.**


moved object
time of day
light switch
waving tree
camouflage
bootstrap
GT
Differenza consecutiva
Media + soglia
Media + std
TAPPMOG

# Approcci per regione

- *Patch* di pixel sono classificate come BG o FG via block-matching
  - cross-correlazione normalizzata
  - confronti tra istogrammi locali
  - in genere, ogni regione ha il proprio modello di BG
    - mediana su una coda di patch
- PRO: più robusti dei modelli per-pixel
- CON: -non precisi spazialmente
  -esosi computazionalmente

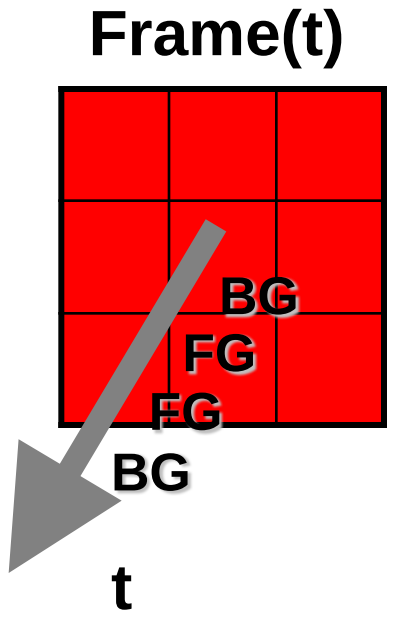


# Approcci per regione

block-matching

# Approcci per frame

- Vengono effettuate elaborazioni sull'intera matrice dei pixel, ed i pixel non sono considerati come processi indipendenti
- in genere, agiscono come supporto per i metodi per pixel o per regione
- vengono applicati per cercare cambi globali della scena che non devono essere identificati come FG – esempio, cambi di illuminazione
- Esempi:
  - Eigenbackgrounds
  - Librerie



Nuria M. Oliver , Barbara Rosario , Alex P. Pentland, A Bayesian Computer Vision System for Modeling Human Interactions, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.22 n.8, p.831-843, August 2000

# Eigenbackgrounds (N. M. Oliver, B. Rosario, and A. P. Pentland, 2000)

- PCA può essere applicato ad una sequenza di n frame per calcolare gli eigenbackgorund
- Buon funzionamento, comparabile a TAPPMOG

# Eigenbackgrounds (N. M. Oliver, B. Rosario, and A. P. Pentland, 2000)

- Gli n frames (meno la loro media) sono organizzati come colonne di una matrice A
- Calcolo la matrice A'A (MxM)
- Estraggo M autovettori $\{\tilde{\mathbf{e}}\}$ , li trasformo negli autovettori per la matrice di covarianza AA' mediante $\mathbf{e} = \mathbf{A}\tilde{\mathbf{e}}$ e li organizzo come colonne di una matrice U
- Quando è disponibile una nuova immagine, z, viene prima sottratta della media, ottenendo $\tilde{\mathbf{z}}$ , che viene proiettata nello spazio a bassa dimensionalità mediante $\boldsymbol{\omega} = \mathbf{U}'\tilde{\mathbf{z}}$ e ricostruita via $\tilde{\mathbf{g}} = \mathbf{U}\boldsymbol{\omega}$
- Calcolo la differenza $\left|\tilde{\mathbf{g}} - \tilde{\mathbf{z}}\right|$ : poiché il sottospazio ben rappresenta solo la parte statica della scena (ossia quello che si è visto di più) la differenza sono solo gli oggetti di foreground

N. Ohta. *A statistical approach to background subtraction for surveillance systems*. In ICCV, pages II: 481--486, Vancouver, Canada, June 2001.

# Librerie

- Si basano su una fase di bootstrapping [Ohta 2001]
  - si addestra un modello per-pixel ordinario
  - si valuta l'errore globale E su ogni frame
  - se l'errore supera una determinata soglia per più di N frame, seleziono quei frame per addestrare un altro modello
  - Ottengo così M modelli, formanti una libreria di modelli
  - Ad ogni time step costruisco la funzione $E_g(t)$:= *numero globale di pixel di FG*
  - Valuto $\frac{\partial E_g(t)}{\partial t}$ : se è oltre una certa soglia, cambiamento brusco della scena
  - In tal caso, seleziono dalla libreria il modello per cui $E_g(t)$ è minimo

# Approcci misti

- Classificano pixel basandosi su valutazioni di regioni locali
- Approcci più in voga
- [Elgammal 2000] - schema

1. Esegui FG test sul valore di pixel i *considerato come segnale independente*

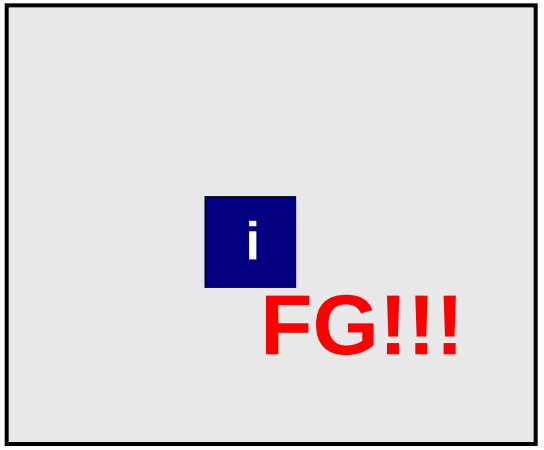
**YES** → 2.

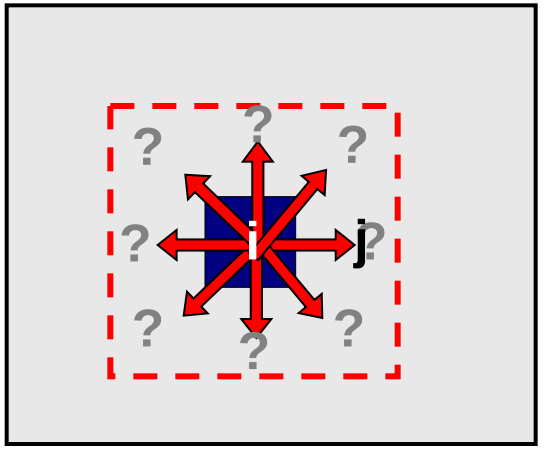
**NO** → i = BG

2. Controlla nei pixel vicini {j} se il valore di pixel i possa essere valutato come valore di BG

**YES** → i = BG

**NO** → i = FG



# Elgammal - risultati

–Vengono minimizzati i falsi negativi

# Spazi colore per la sottrazione del BG

- Il setting più semplice (e veloce computazionalmente) è quello di avere sequenze a livelli di grigio
  - Poco preciso (vedasi mascheramento del FG)
- Considerando input a colori, i risultati, naturalmente, migliorano. Gli spazi colore che si considerano abitualmente sono
  - RGB [Wallflower 2001]
  - RGB normalizzato

$$\tilde{R} = \frac{R}{R+G+B} \qquad \tilde{G} = \frac{G}{R+G+B} \qquad \tilde{B} = \frac{B}{R+G+B}$$

meno sensibile a leggeri cambi di illuminazione, ma anche meno sensibile alla *lightness*, ossia alla differenza tra bianco-grigio-nero

  - HSV utilizzato soprattutto per l'eliminazione delle ombre (a seguire)

# Spazi colore per la sottraizone del BG

- Nel caso di RGB e RGB normalizzato ogni canale può essere valutato in maniera indipendente (anche se non è corretto)
  - Dato $FG_X$ il foreground trovato nel canale x, il FG finale è

$$FG = \vee\left(FG_R, FG_G, FG_B\right)$$

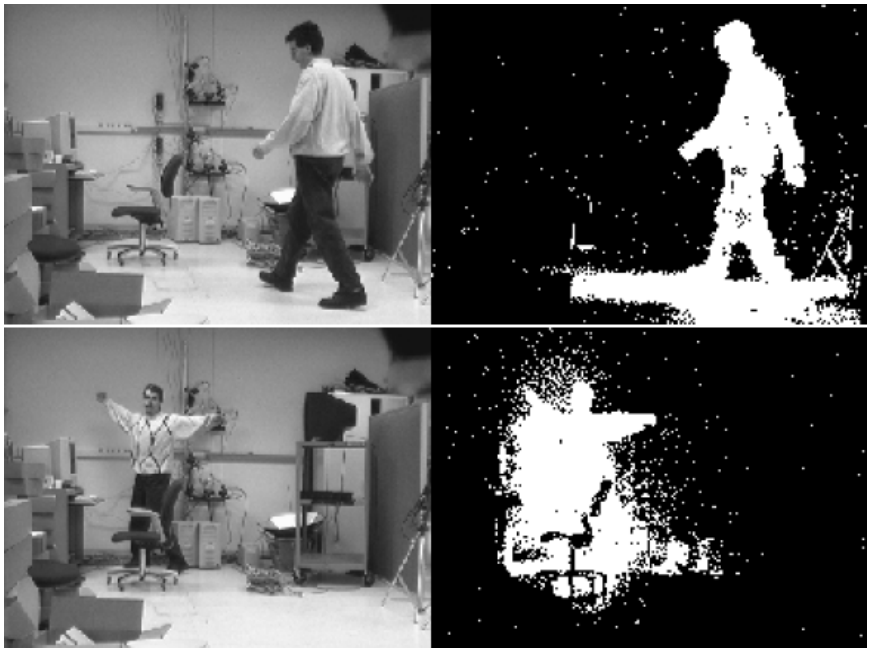
- Alternativamente, un singolo valore di pixel puo' essere modellato come unico segnale a covarianza diagonale (anche questa è una semplificazione scorretta teoricamente – c'è dipendenza tra i singoli canali colore – ma funzionalmente vantaggiosa)

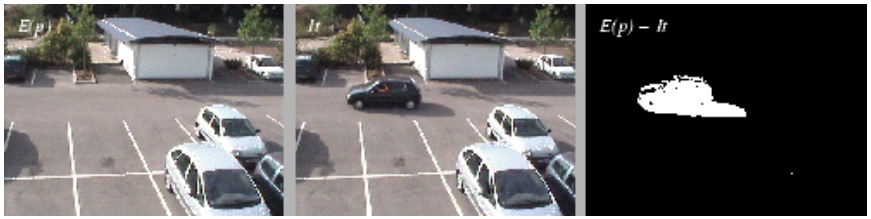
$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_R^2 & 0 & 0 \\ 0 & \sigma_G^2 & 0 \\ 0 & 0 & \sigma_B^2 \end{bmatrix}$$

# Il problema delle ombre

- Problema grave: soprattutto nelle interfacce di HCI è basilare acquisire la forma esatta dell'oggetto in movimento

Esempi:

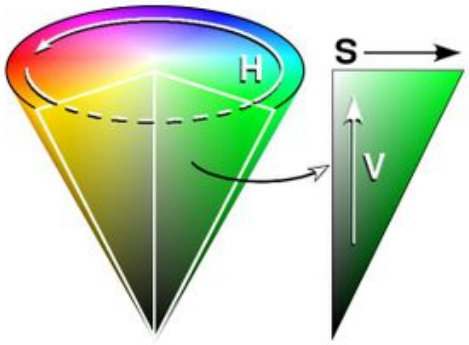




R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, Ghosts, and shadows in video streams," IEEE Trans. Pattern Anal. Mach. Intell. 25,1337-1342 (2003)

# Il problema delle ombre

- La maggior parte delle soluzioni lavora utilizzando delle soglie sullo spazio colore HSV
- Approccio semplice ma effettivo [Cucchiara 2003]

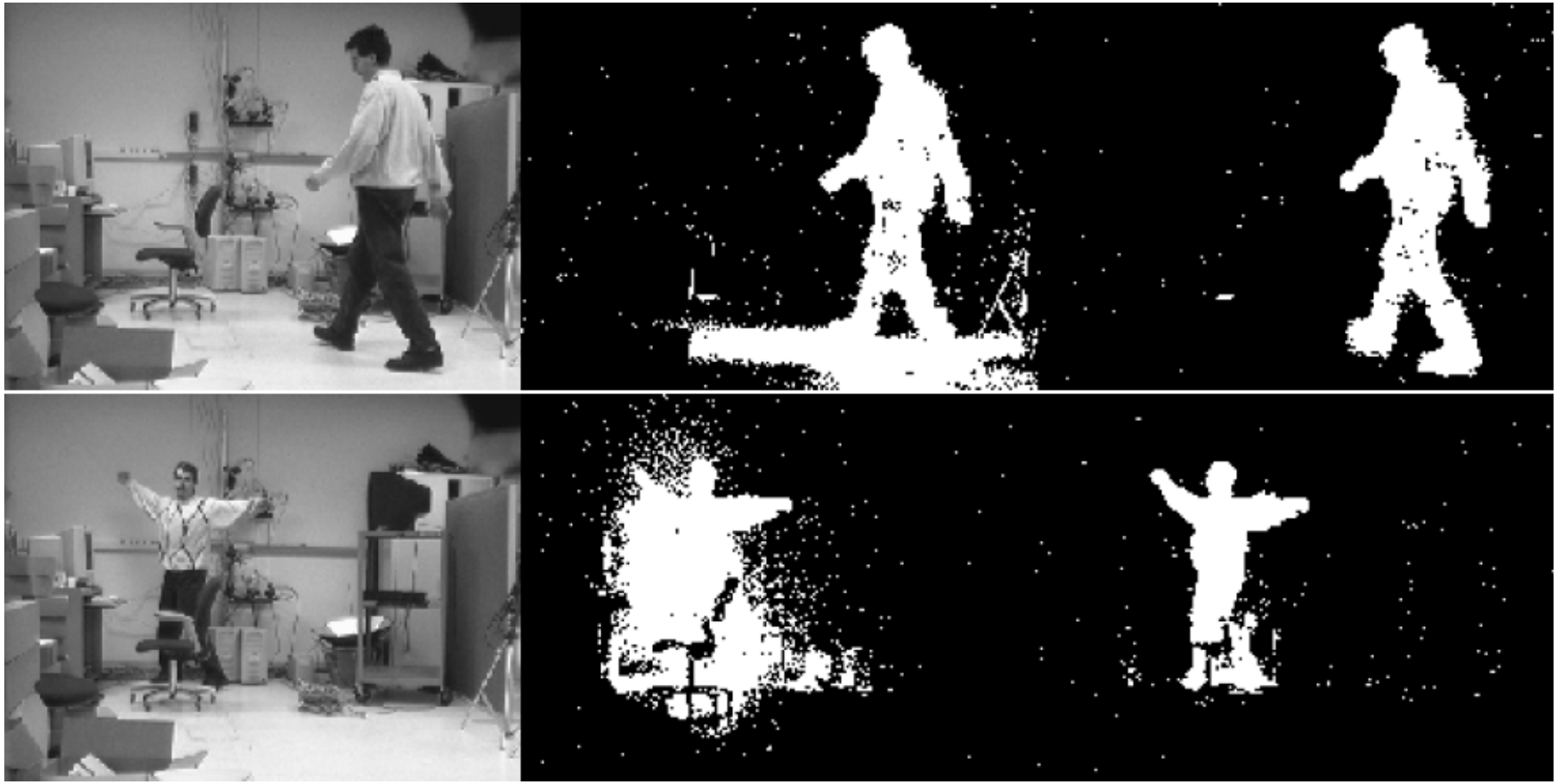



$$S_i^{(t)} = \begin{cases} 1 & \text{se } \alpha \le \dfrac{z_i^{(t)}.V}{b_i^{(t)}.V} \le \beta \wedge \left| z_i^{(t)}.S - b_i^{(t)}.S \right| \\ & \le \tau_S \wedge D_i^{(t)}.H \le \tau_H \quad \alpha \in [0,1], \beta \in [0,1] \\ 0 & \text{altrimenti} \end{cases}$$

flag di ombra

$$D_i^{(t)}.H = \min\left( \left| z_i^{(t)}.H - b_i^{(t)}.H \right|, 360 - \left| z_i^{(t)}.H - b_i^{(t)}.H \right| \right)$$

# Il problema delle ombre



# “Retro”-applicazioni

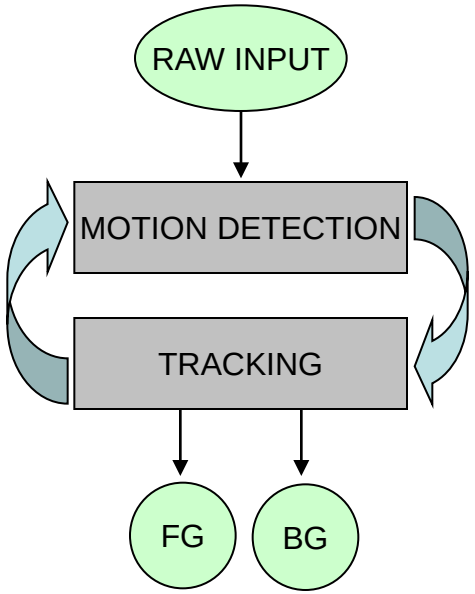
- Motion detection come pre-processing oer
  - face detection
    - dopo motion detection→metodi di proiezione
  - face recognition
    - risulta più semplice fare allineamento
    - risultati migliori dell’operazione di masking

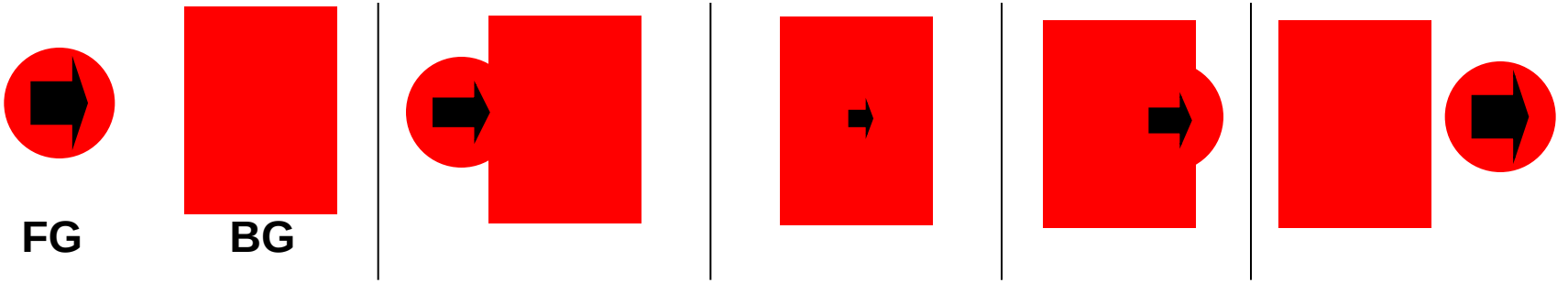
# Nuove sequenze “difficili” e di test

**TAPPMOG**

# Considerazioni finali

- Non può esistere un metodo infallibile di background subtraction
- In un metodo di sottrtazione del BG, si modellano la formazione e l'updating di un modello di background, ignorando il FG
- Spesso il FG ha le stesse caratteristiche del BG → problema
- Soluzione: si connettono in feedback approcci per la modellazione del FG→ tracking (modellano l'appearance e la dinamica di un oggetto di FG)
- la dinamica è discriminante




(la grandezza della freccia è proporzionale al peso dato alla dinamica nella definizione del FG)

# Materiale aggiuntivo

- Review di metodi di sottrazione del BG
  - Piccardi.pdf
- Rimozione delle ombre
  - Cucchiara03detecting.pdf
- TAPPMOG
  - vsam-pami-tracking.pdf