Background Subtraction

Background Subtraction

- Given an image (mostly likely to be a video frame), we want to identify the foreground objects in that image!

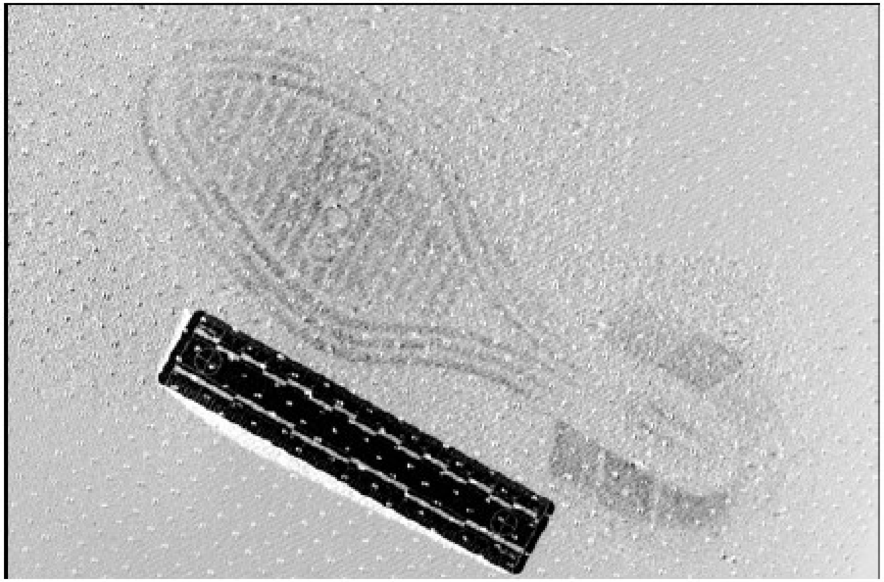
$\Rightarrow$

Motivation

- In most cases, objects are of interest, not the scene.
- Makes our life easier: less processing costs, and less room for error

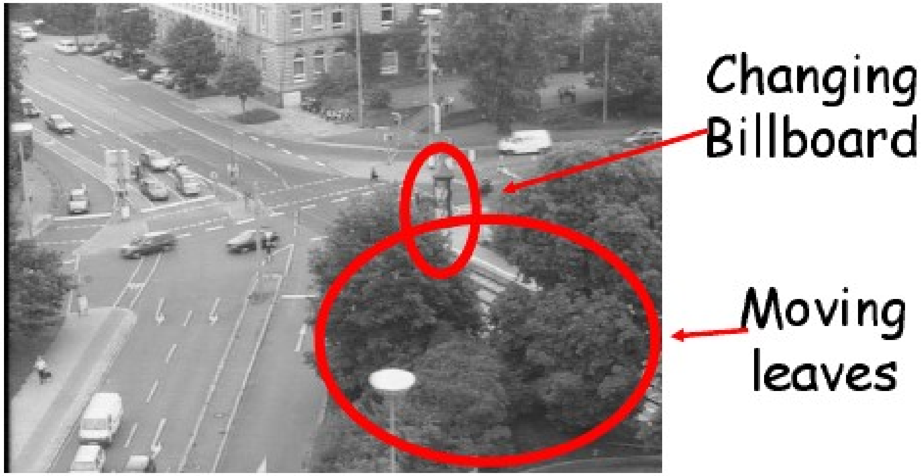
Widely Used!

- Traffic monitoring (counting vehicles, detecting & tracking vehicles),
- Human action recognition (run, walk, jump, squat, . . .),
- Human-computer interaction ("human interface"),
- Object tracking (watched tennis lately?!?),
- And in many other cool applications of computer vision such as digital forensics.



Requirements

- A reliable and robust background subtraction algorithm should handle:
  - Sudden or gradual illumination changes,
  - Long-term scene changes (a car is parked for a month).
  - high frequency, repetitive motion in the background (such as tree leaves, flags, waves, . . .)



Requirements

- ...continues
  - Secondary illumination effects (e.g. shadows cast by foreground objects)

Simple Approach

1. Estimate the background for time t.
2. Subtract the estimated background from the input frame.
3. Apply a threshold T to the absolute difference to get the foreground mask.

Image at time t

-

Background at time t

But, how can we estimate the background?

Frame Differencing

- Background is estimated to be the previous frame.
- Background subtraction equation then becomes:

$$B(x, y, t) = I(x, y, t-1)$$

$$|I(x, y, t) - I(x, y, t-1)| > T$$

- Depending on the object structure, speed, frame rate and global threshold, this approach may or may not be useful (usually not).

-

Frame Differencing

T=25

T=50

T=100

T=200

Mean Filter

- In this case the background is the mean of the previous n frames:

$$B(x,y,t) = \frac{1}{n}\sum_{i=0}^{n-1} I(x,y,t-i)$$

$$\left| I(x,y,t) - \frac{1}{n}\sum_{i=0}^{n-1} I(x,y,t-i) \right| > T$$

n=10

Estimated background

Estimated foreground

Mean Filter

n=20

Estimated background

Estimated foreground

n=50

Estimated background

Estimated foreground

Median Filter

- Assuming that the background is more likely to appear in a scene, we can use the median of the previous n frames as the background model:

$$B(x,y,t) = \underset{i=0\ldots n-1}{\text{median}} \big(I(x,y,t-i)\big)$$

$$\left| I(x,y,t) - \underset{i=0\ldots n-1}{\text{median}} \big(I(x,y,t-i)\big) \right| > T$$

n=10

Estimated background

Estimated foreground

Median Filter

n=20

Estimated background

Estimated foreground

n=50

Estimated background

Estimated foreground

Advantages vs. Shortcomings

- Advantages:
  - Extremely easy to implement and use!
  - All pretty fast.
  - Corresponding background models are not constant, they change over time.

- Disadvantages:
  - Accuracy of frame differencing depends on object speed and frame rate!
  - Mean and median background models have relatively high memory requirements.
    - In case of the mean background model, this can be handled by a running average

Advantages vs. Shortcomings

- There is another major problem with these simple approaches:

  1. There is one global threshold, Th, for all pixels in the image.

  2. And even a bigger problem:
  this threshold is not a function of t.

- So, these approaches will not give good results in the following conditions:

  - if the background is bimodal,
  - if the scene contains many, slowly moving objects (mean & median),
  - if the objects are fast and frame rate is slow (frame differencing),
  - and if general lighting conditions in the scene change with time!

Early Approaches

Schemes	Background modeling	Foreground detection
Frame Differencing	$B_t = I_{t-1}$	Foreground candidate if $\frac{\lvert I_t(x,y)-B_t(x,y)-\mu_t \rvert}{\sigma_t} > \Gamma$
Kalman Filter	$B_t = B_{t-1} + \begin{cases} \alpha_{small}\cdot(I_t-B_{t-1}), \text{ if } F_{t-1} = 1 \\ \alpha_{large}\cdot(I_t-B_{t-1}), \text{ otherwise} \end{cases}$	
Adaptive Median	$B_t = B_{t-1} + \begin{cases} -1, \text{ if } I_t \leq B_{t-1} \\ 1, \text{ otherwise} \end{cases}$	
Median	$B_t = \text{median } \{I_{t-T+1}, I_{t-T+2}, \ldots, I_t\}$	

Gaussian Model

- C. Stauffer and W.E.L. Grimson “Adaptive Background Mixture Models for Real-Time Tracking”

- Model the values of a particular pixel as a mixture of adaptive Gaussians.
  - Why mixture? Multiple surfaces appear in a pixel.
  - Why adaptive? Lighting conditions change.

- At each iteration Gaussians are evaluated using a simple heuristic to determine which ones are mostly likely to correspond to the background.
- Pixels that do not match with the “background Gaussians” are classified as foreground.
- Foreground pixels are grouped using 2D connected component analysis.

Online Mixture Model

- At any time t, what is known about a particular pixel (x0 , y0) is its history:

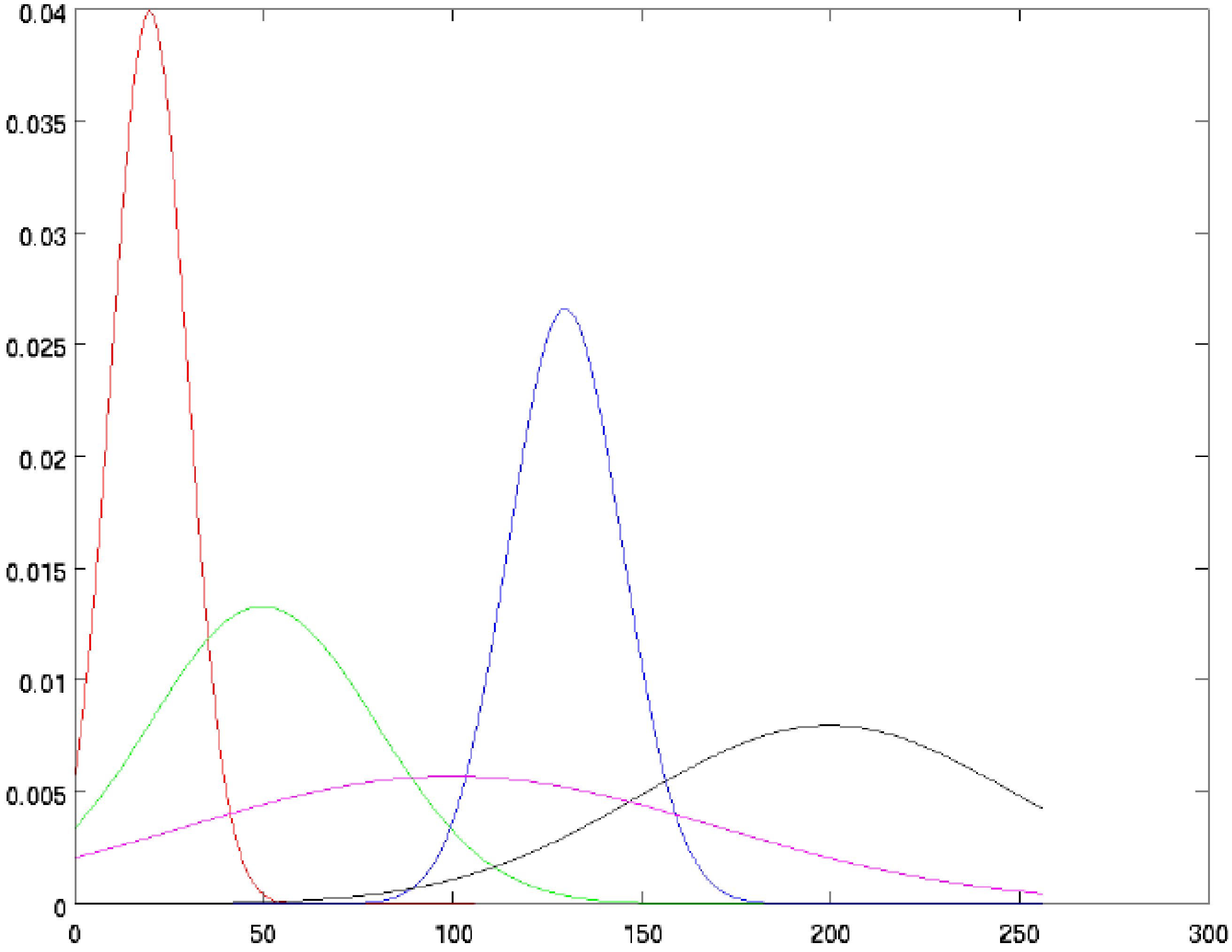
$$\{X_1, \ldots, X_t\} = \{I(x_0, y_0, i) | 1 \leq i \leq t\}$$

- This history is modeled by a mixture of K Gaussian distributions:

$$P(X_t) = \sum_{i=1}^{K} \omega_{it} \mathcal{N}(X_t | \mu_{it}, \Sigma_{it})$$

$$\mathcal{N}(X_t | \mu_{it}, \Sigma_{it}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma_{it}|} \exp\left(-\frac{1}{2}(X_t - \mu_{it}^T \Sigma_{it}^{-1} (X_t - \mu_{it})\right)$$

Online Mixture Model



Model Adaptation

- An on-line K-means approximation is used to update the Gaussians.
- If a new pixel value, X_{t+1} , can be matched to one of the existing Gaussians (within 2.5σ), that Gaussian's $\mu_{i,t+1}$ and $\sigma^2_{i,t+1}$ are updated as follows:

$$\mu_{it+1} = (1-\rho)\mu_{it} + \rho X_{t+1}$$
$$\sigma^2_{it+1} = (1-\rho)\sigma^2_{it} + \rho(X_{t+1} - \mu_{it+1})^2$$

with $\rho = \alpha\mathcal{N}(X_{t+1)|\mu_{ti},\sigma^2_{it}})$

- Prior weights of all Gaussians are adjusted as follows:

$$\omega_{it+1} = (1-\alpha)\omega_{it} + \alpha M_{it+1}$$

- Where $M_{i,t+1}$=1 for the matching Gaussian, 0 for all the others

Model Adaptation

- If X_{t+1} do not match to any of the K existing Gaussians, the least probably distribution is replaced with a new one.
  - Warning!!! “Least probably” in the ω/σ sense (will explain in a second)
  - New distribution has $\mu_{t+1} = X_{t+1}$, a high variance and a low prior weight.

Background Model Estimation

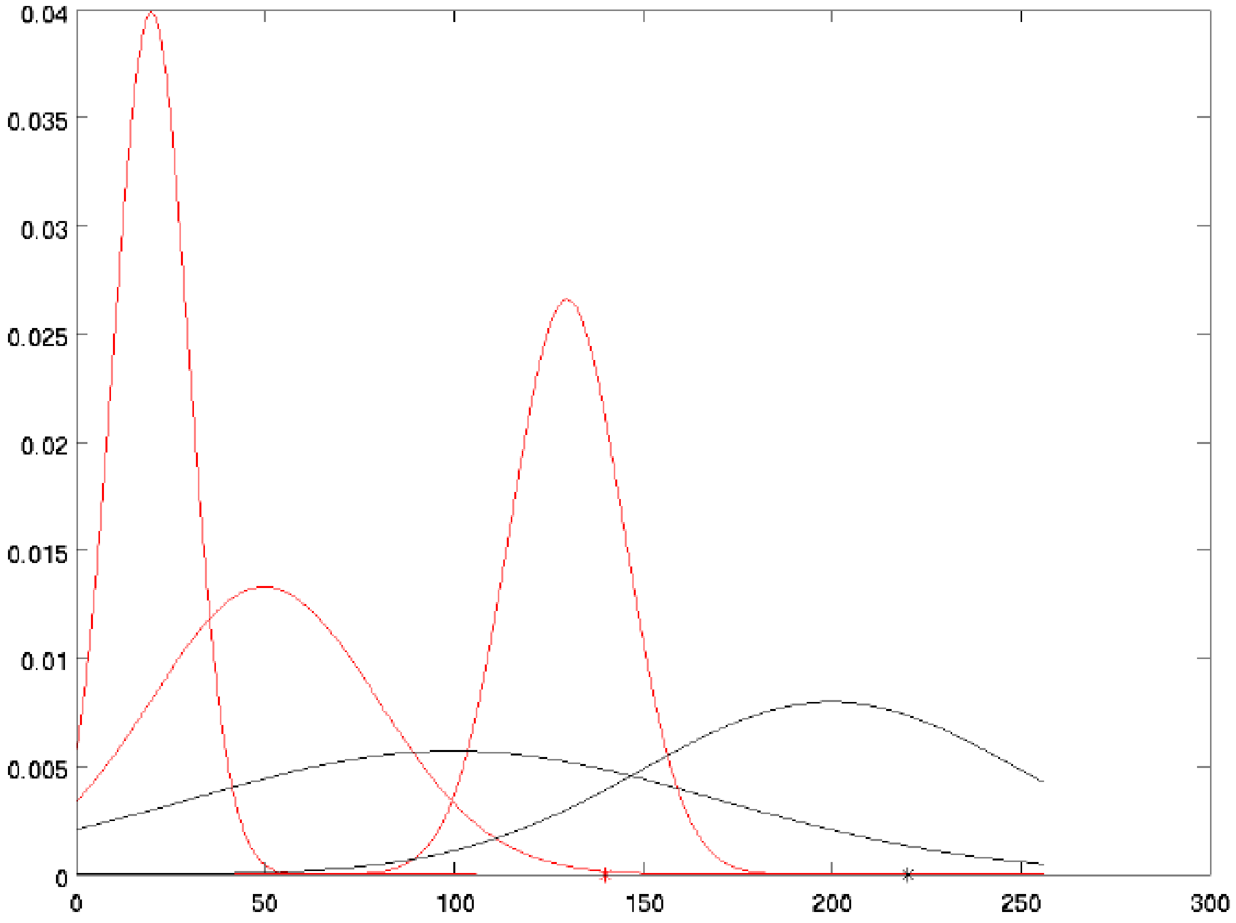
- Heuristic: the Gaussians with the most supporting evidence and least variance should correspond to the background (Why?).
- The Gaussians are ordered by the value of ω/σ (high support & less variance will give a high value).
- Then simply the first B distributions are chosen as the background model:

$$B = \underset{b}{\operatorname{argmin}} \left(\sum_{i=1}^{b} \omega_i > T \right)$$

where T is minimum portion of the image which is expected to be background.

Background Model Estimation

- After background model estimation red distributions become the background model and black distributions are considered to be foreground.



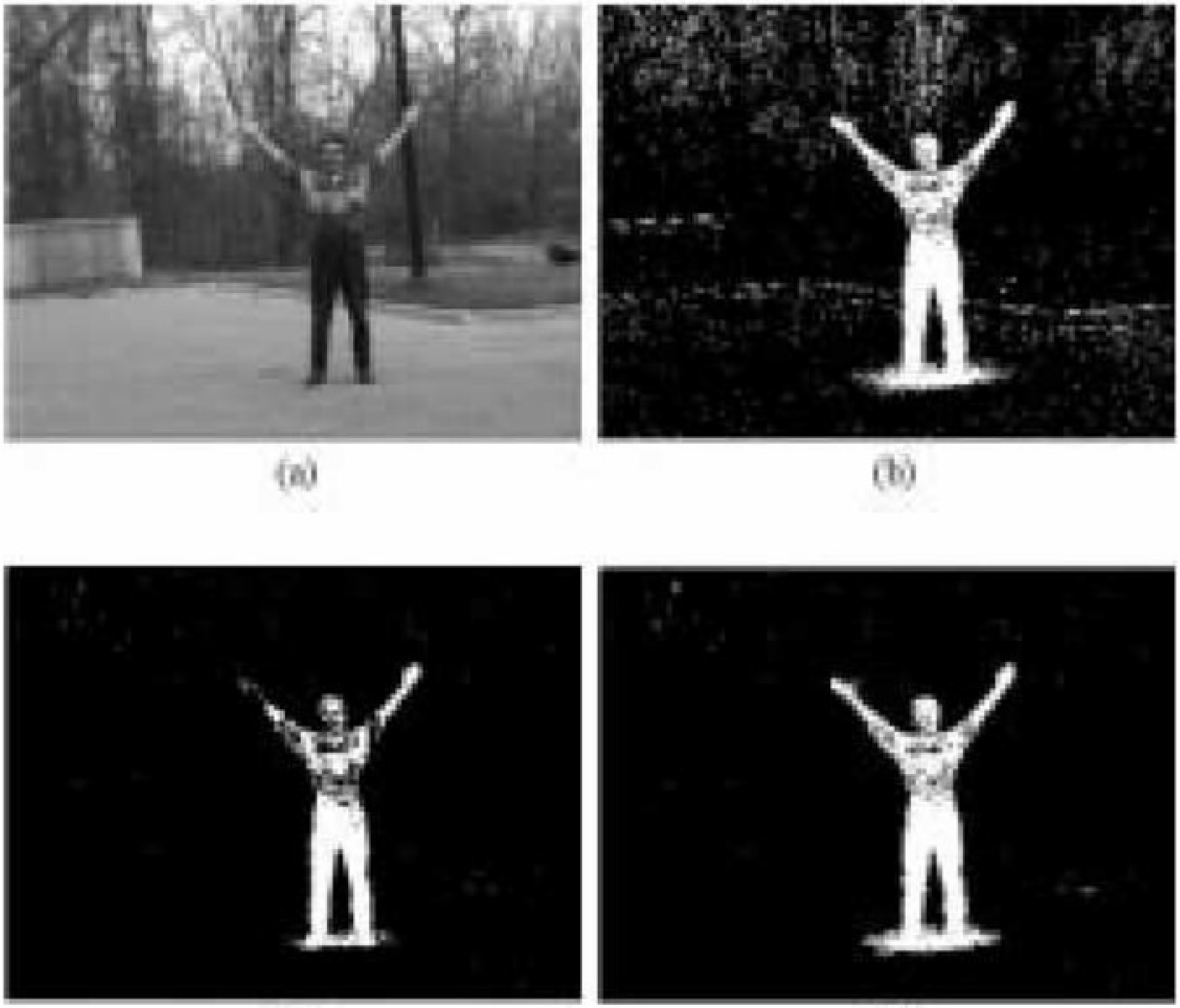
Advantages vs. Shortcomings

- Advantages:
  - A different "threshold" is selected for each pixel.
  - These pixel-wise "thresholds" are adapting by time.
  - Objects are allowed to become part of the background without destroying the existing background model.
  - Provides fast recovery.

- Disadvantages:
  - Cannot deal with sudden, drastic lighting changes!
  - Initializing the Gaussians is important (median filtering).
  - There are relatively many parameters, and they should be selected intelligently.

Post Processing

- Erosion and dilation



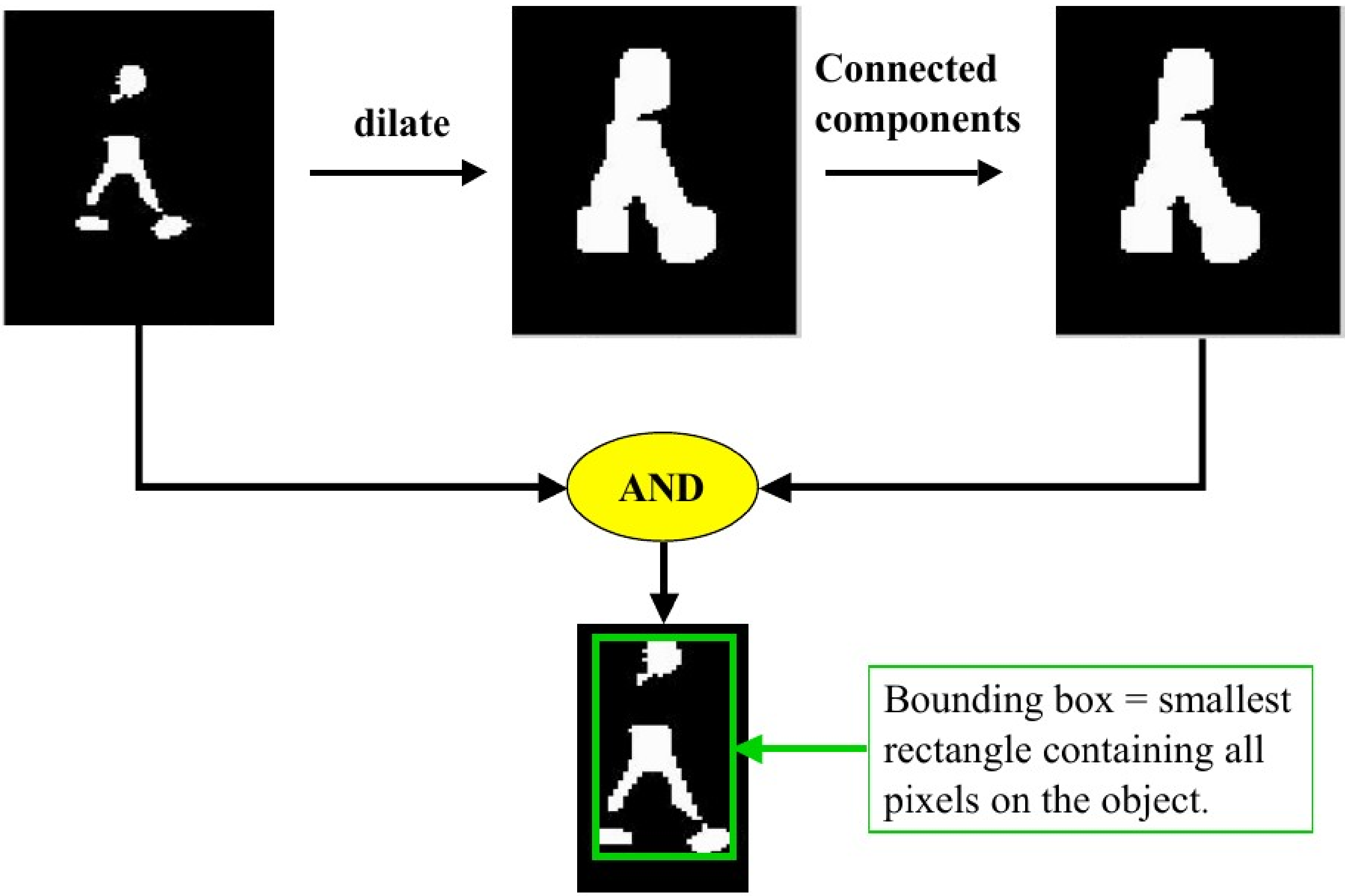
(a) (b) (c) (d)

Removal of shadows

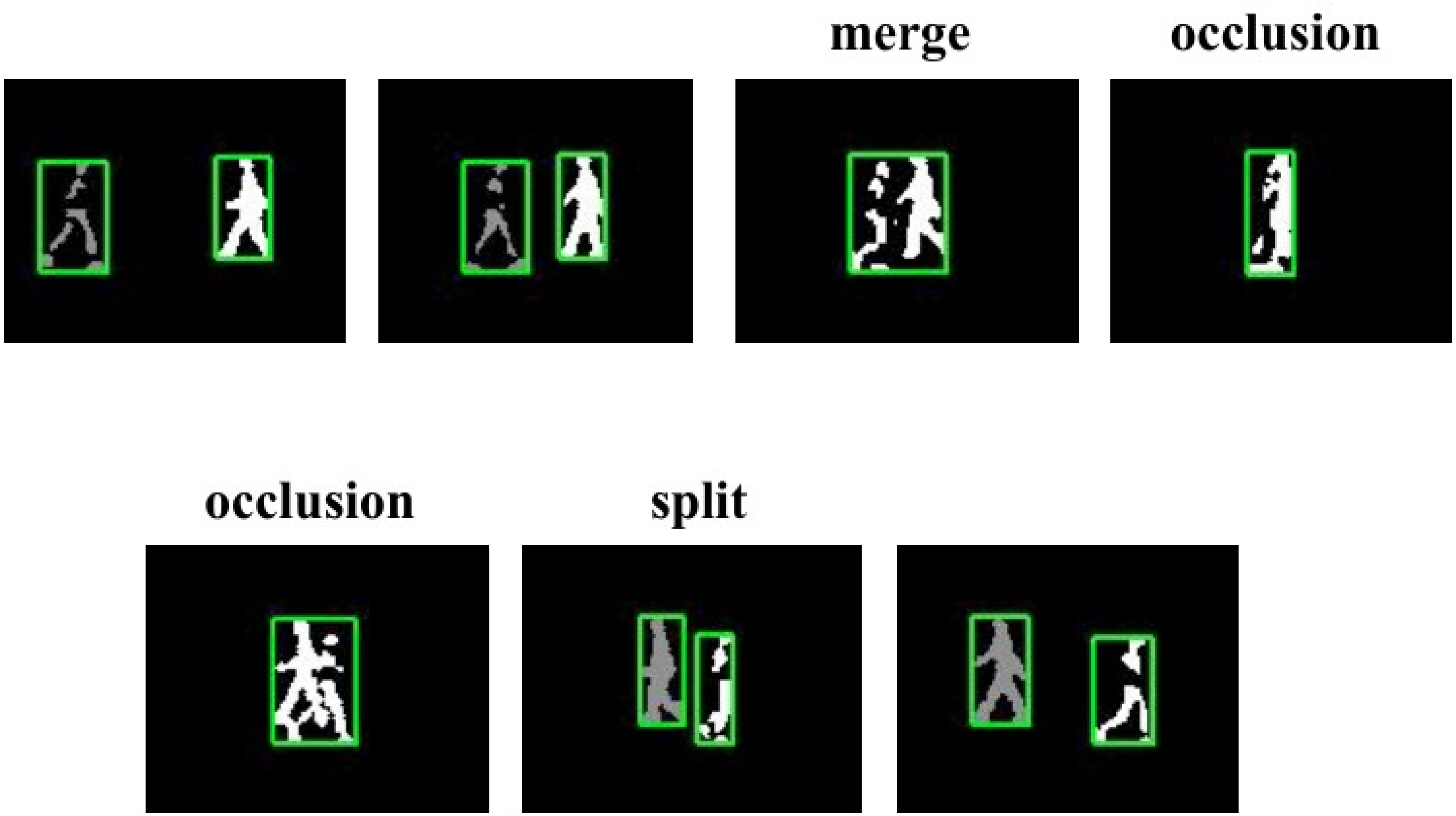
- Shadows change luminance but not chromaticity

Grouping Pixels into Blobs

- median filter to remove noisy pixels
- connected components (with gap spanning)
- Size filter to remove small regions

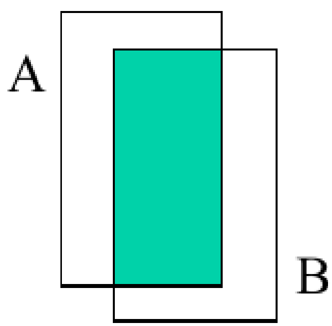


Blob Merge and Split



Data Association

- Determining the correspondence of blobs across frames is based on feature similarity between blobs.
- Commonly used features: location , size / shape, velocity, appearance
- For example: location, size and shape similarity can be measured based on bounding box overlap:

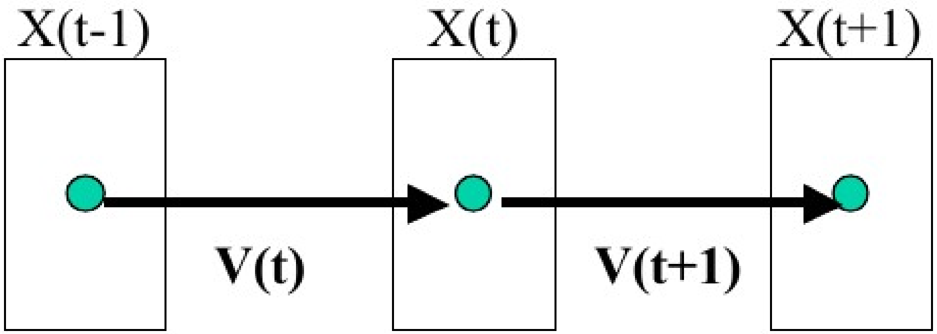


$$\text{score} = \frac{2 * \text{area(A and B)}}{\text{area(A)} + \text{area(B)}}$$

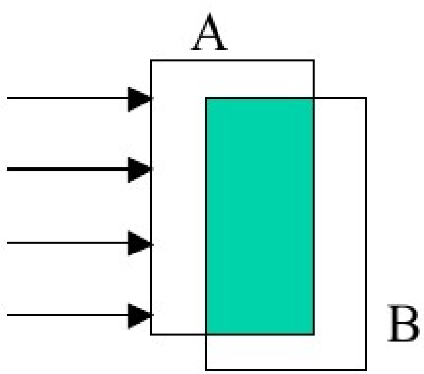
A = bounding box at time t
B = bounding box at time t+1

Data Association (Velocity)

- It is common to assume that objects move with constant velocity



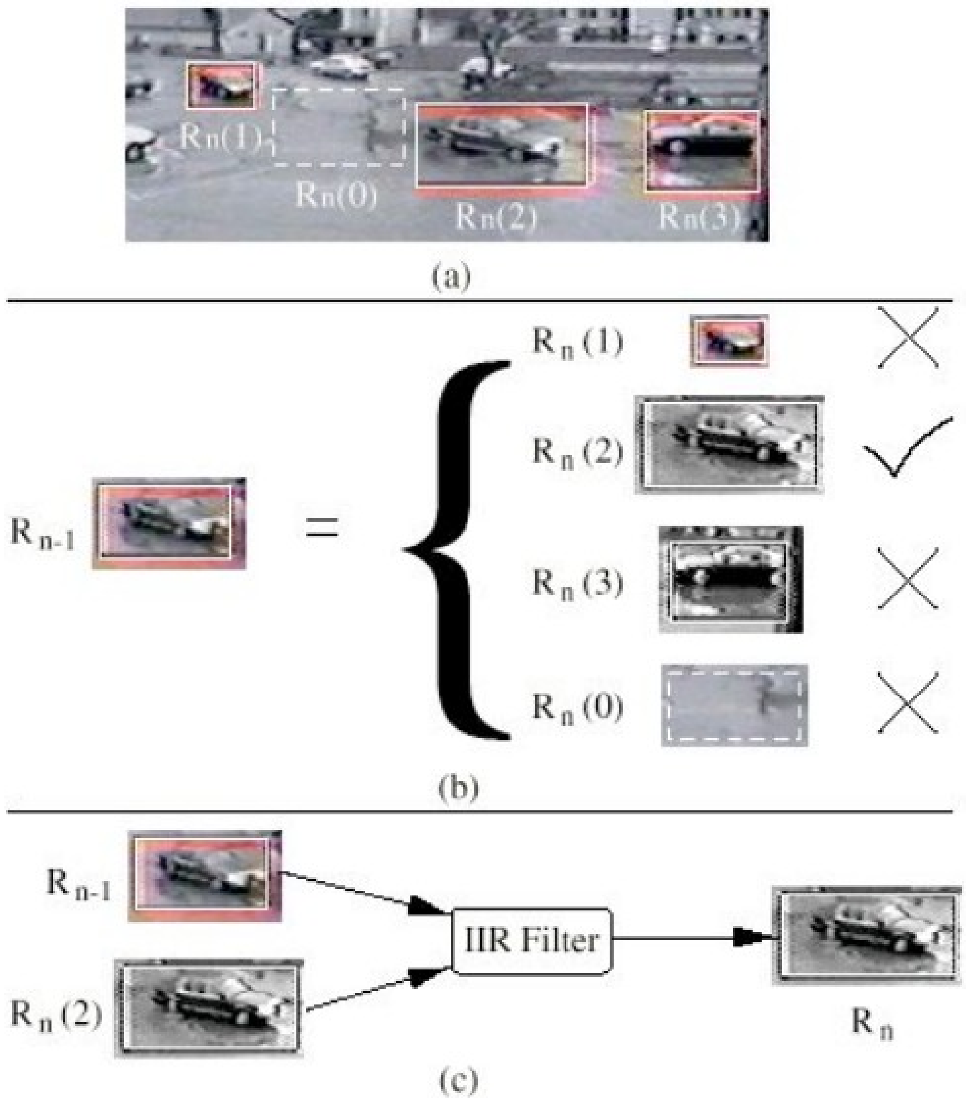
constant velocity assumes V(t) = V(t+1)



$$\text{score} = \frac{2 * \text{area}(A \text{ and } B)}{\text{area}(A) + \text{area}(B)}$$

A = bounding box at time t, adjusted by velocity V(t)
B = bounding box at time t+1

Data Association (Appearance)

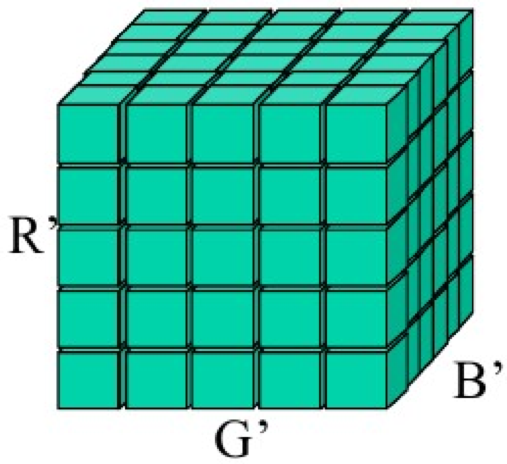


Extract blobs

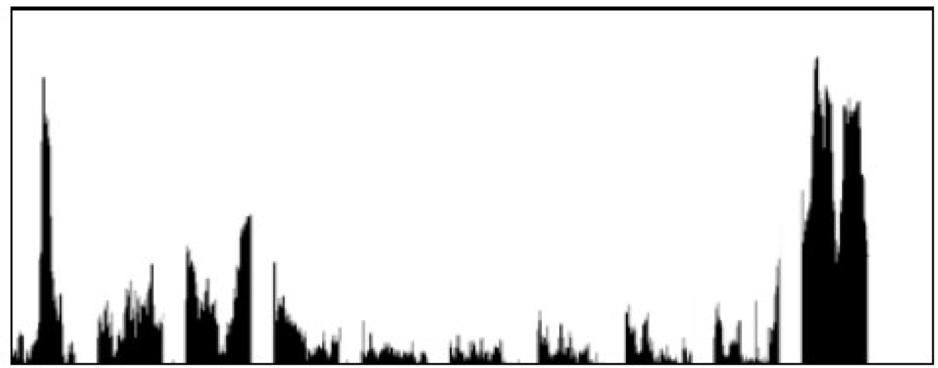
Data association via normalized correlation.

Update appearance template of blobs

Appearance via Color Histograms



discretize


Color distribution (1D histogram normalized to have unit weight)

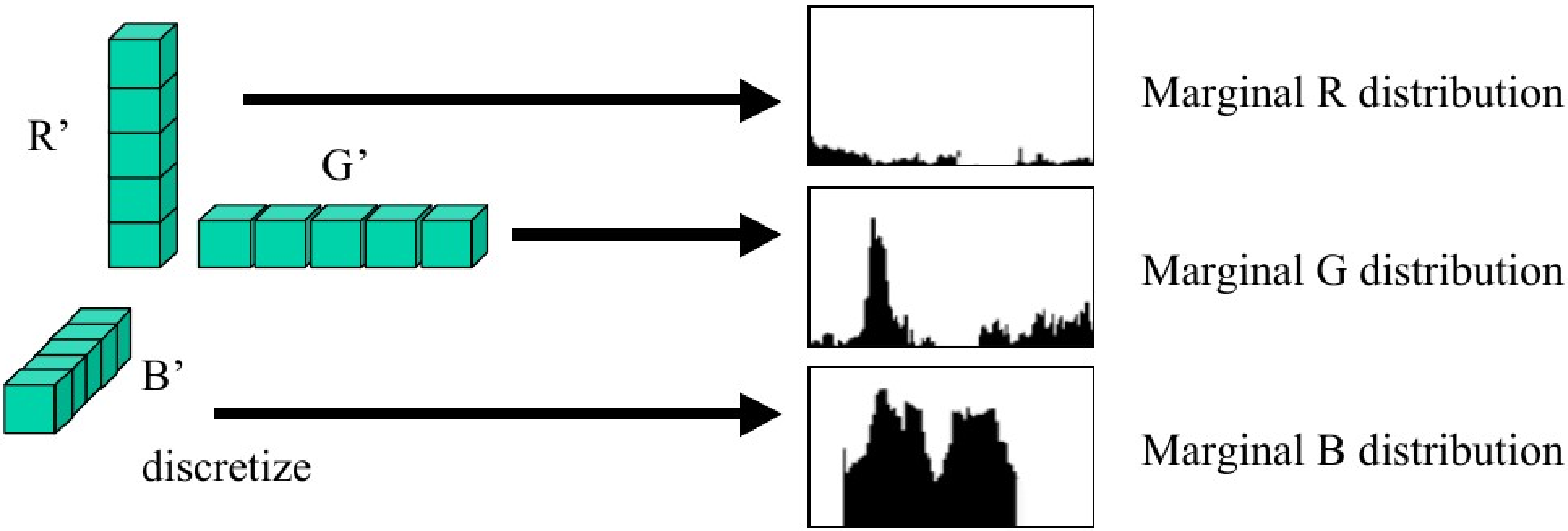
R’ = R << (8 - nbits)
G’ = G << (8 - nbits)
B’ = B << (8 - nbits)

Total histogram size is (2^(8-nbits))^3

example, 4-bit encoding of R,G and B channels yields a histogram of size 16*16*16 = 4096.

Appearance via Reduced Histograms

- Histogram information can be much much smaller if we are willing to accept a loss in color resolvability.

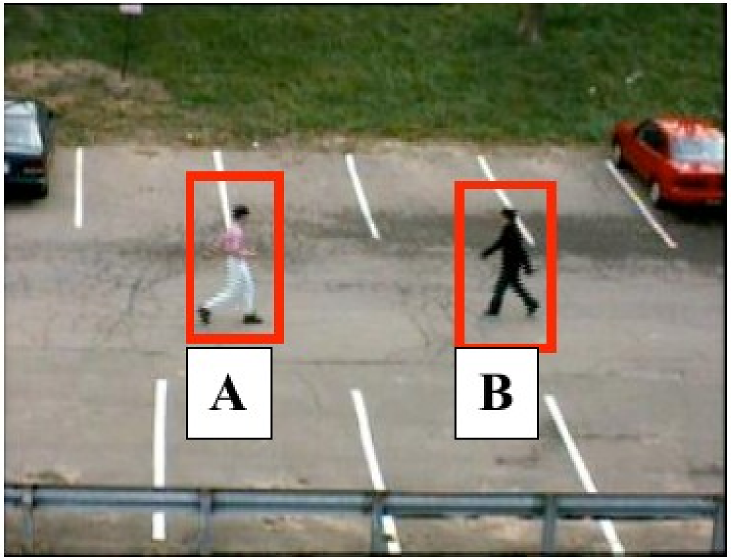


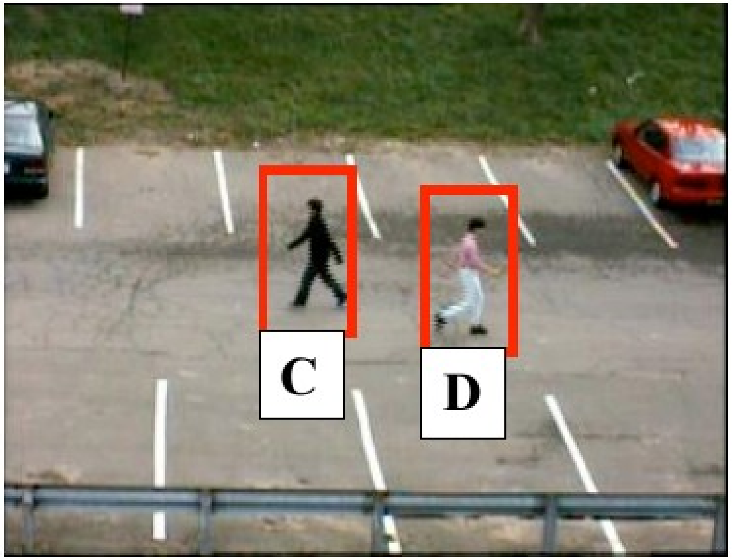
R' = R << (8 - nbits)
G' = G << (8 - nbits)
B' = B << (8 - nbits)

Total histogram size is 3*(2^(8-nbits))

example, 4-bit encoding of R,G and B channels yields a histogram of size 3*16 = 48.

Association after Merge and Split





$\Delta(A,C) = 2.03$

$\Delta(A,D) = 0.39$ ●   **A -> D**

$\Delta(B,C) = 0.23$ ●   **B -> C**

$\Delta(B,D) = 2.0$