Master in
Computer Vision
*Barcelona*

**Module:** Video Analysis

**Lecture 2:** Video segmentation

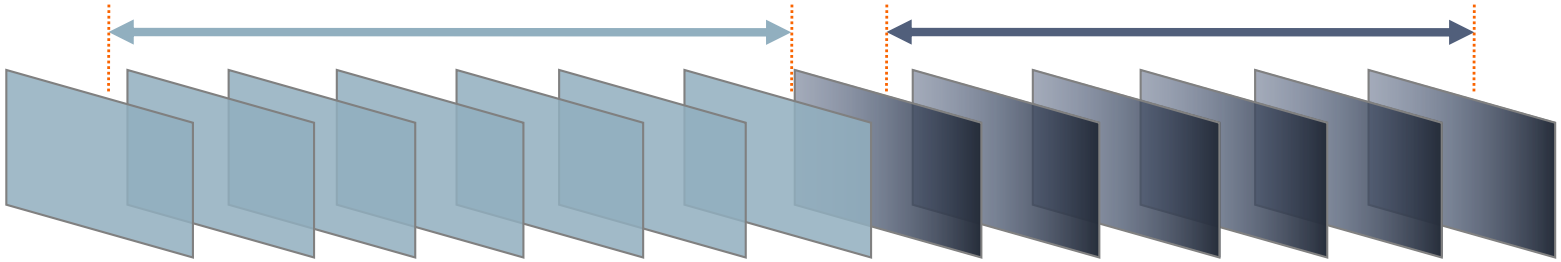
**Lecturer:** Montse Pardàs

# Outline

- Introduction to video segmentation
- Shot segmentation
- Moving object segmentation
    - Introduction
    - Still background estimation
    - Variable background estimation: Single Gaussian
    - Variable background estimation: Multiple Gaussians
    - Other change detection techniques
  - Shadow detection
  - Connected component analysis and tracking
- Region segmentation
    - Spatial segmentation and tracking
    - Spatio-temporal segmentation

# Introduction to video segmentation

- **Different meanings**:
  - **Shot detection**: The segmentation is seen as a 1D (temporal) problem.
  - A **temporal segmentation** is performed, dividing the video into segments that should be related to the different shots in the sequence.



# Introduction to video segmentation

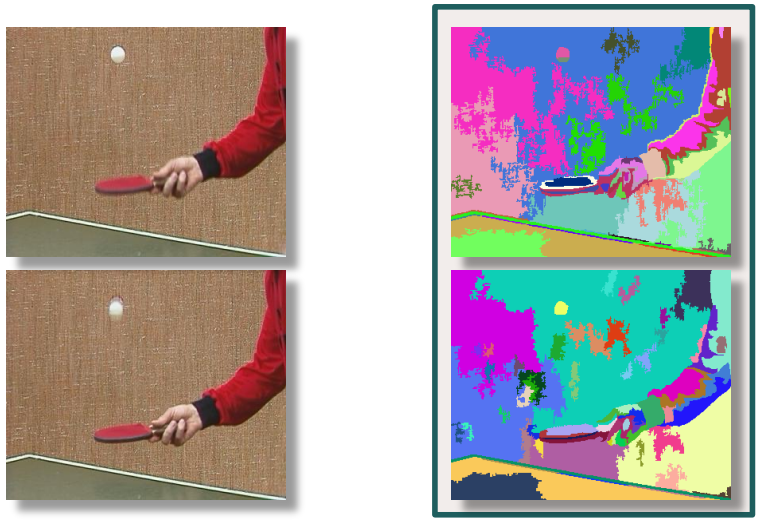
- **Different meanings**:

  **Moving object segmentation**: Detect and segment moving objects from a video sequence of a fixed camera
  - Background – static scene
  - Foreground – moving objects



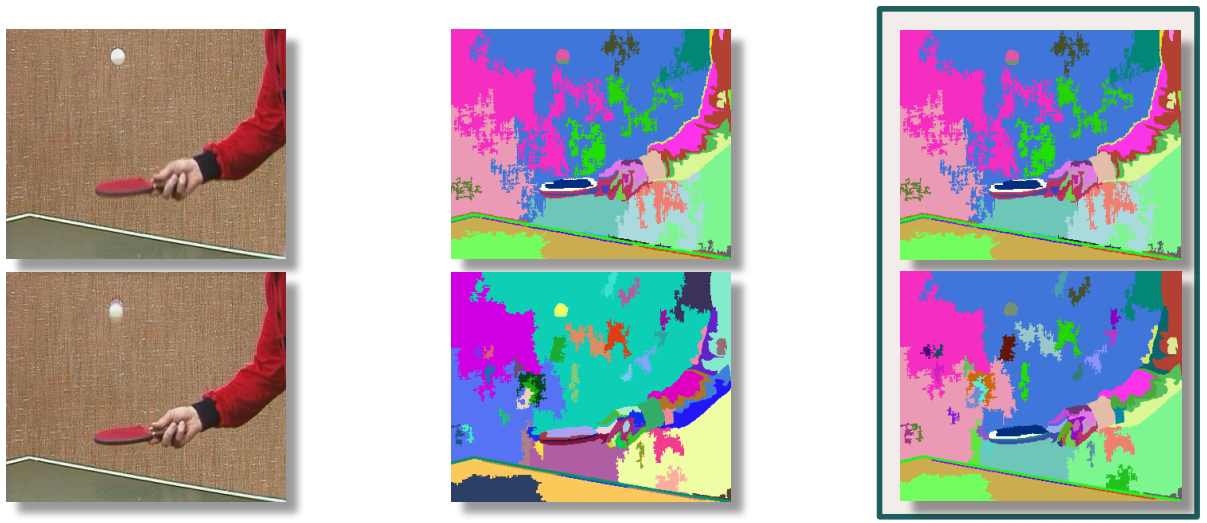
# Introduction to video segmentation

- **Different meanings**:
  - **2D Region segmentation**: The sequence is understood as a set of isolated images. **Spatial segmentation** is performed on each separated image (2D signal) in the sequence. Regions have to be related to the objects in the scene.
    - It is the same problem as the **image segmentation** problem.
    - Object labels in consecutive images **may not be coherent**.



# Introduction to video segmentation

- **Different meanings**:
  - **3D region segmentation**: The sequence is understood as a set of temporally related images. **Spatial/Temporal segmentation** is performed on the sequence (3D signal). Regions have to be related to the objects in the scene and their temporal evolution.
    - Direct extension of the **image segmentation** problem to the 3D case.
    - Object labels in consecutive images **should be coherent**.



# Introduction to video segmentation

- The division in three steps proposed for image segmentation can be adopted:


- **Remove useless or annoying information**
- **Preserve shape information**

- Elements belonging to the same regions should correspond to **homogeneous features**

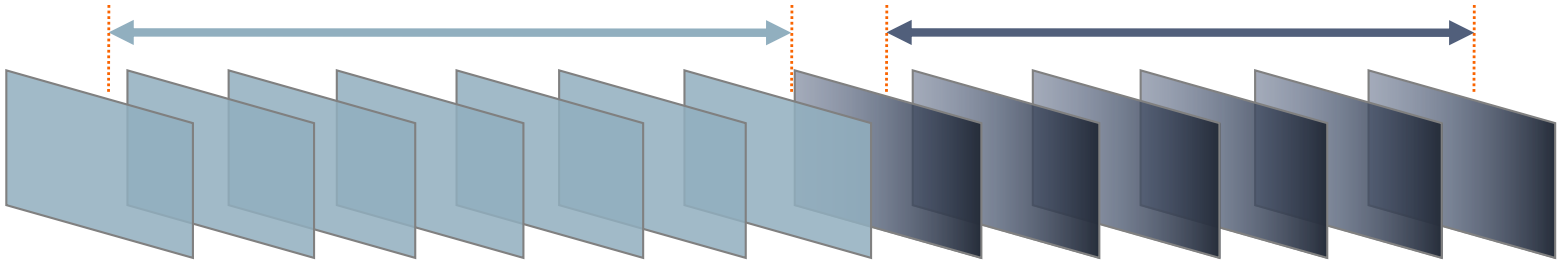
- **Feature space**:
  - Gray level/Color
  - Texture
  - Depth
  - Motion
  - Frame difference
  - DFD
  - Histogram

- **Decision space**:
  - 1D: Temporal
  - 2D: Spatial
  - 3D: Spatial/temporal
- **Partition definition**:
  - ✓ Transition
  - ✓ Homogeneity

# Outline

- Shot segmentation
- Moving object segmentation
- Region segmentation

# Shot segmentation (I)

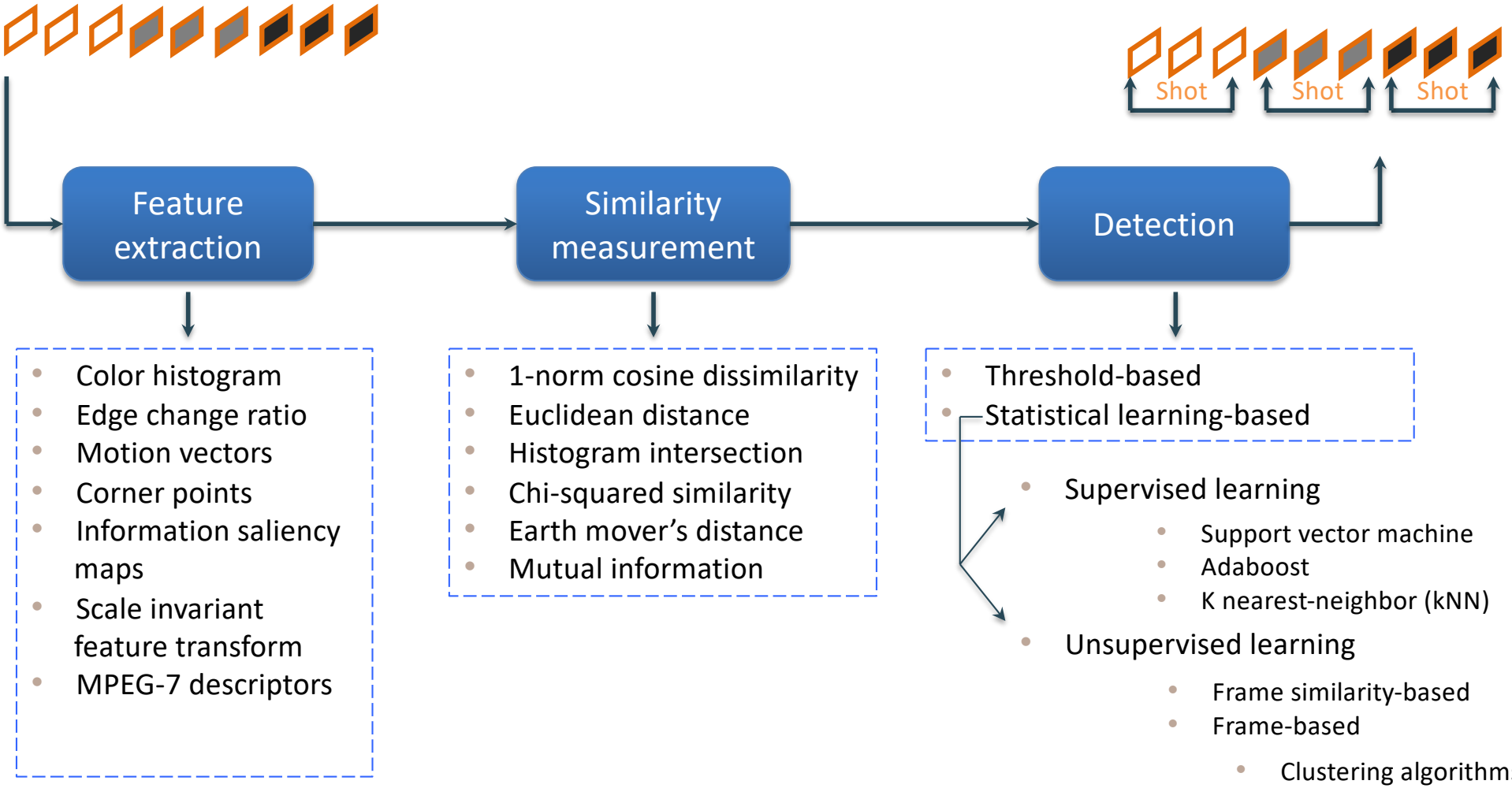


- A **video shot** is a sequence of frames captured by one camera in a single continuous action in time and space.


- **Feature extraction and decision**
  - Frame difference (FD)
  - Frame histogram comparison
  - Displaced frame difference (DFD)

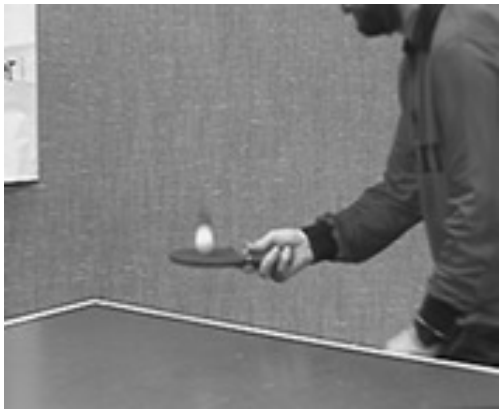
Temporal gradients

# Shot segmentation (I)



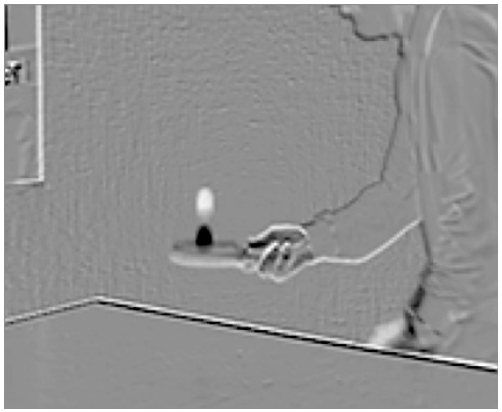

W. Hu, A survey on visual content-based video indexing and retrieval
IEEE Transactions on Systems, November 2011

# Shot segmentation

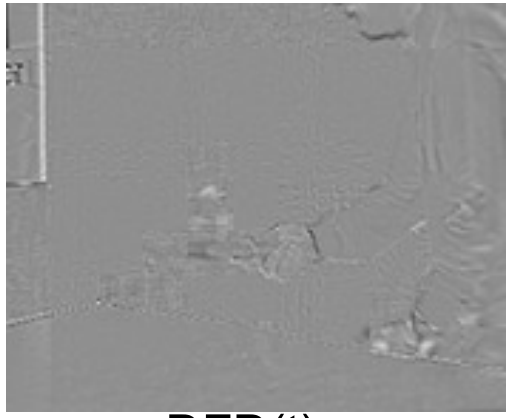


I(t-1)

I(t)



FD(t)



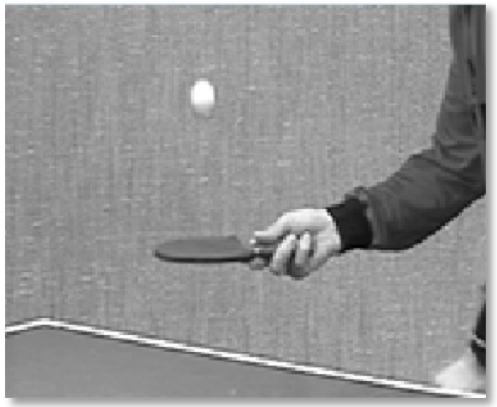
DFD(t)

# 2D Motion analysis

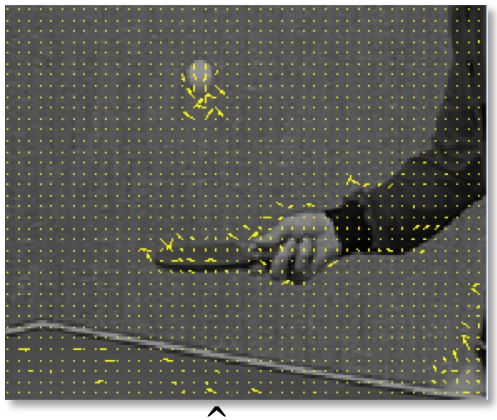
- In practice, the optical flow has to be estimated and may differ from the ideal one.

- If $\hat{\vec{D}}(\vec{r})$ denotes the estimated optical flow, the reference image to which the optical flow is applied is called the motion compensated image.

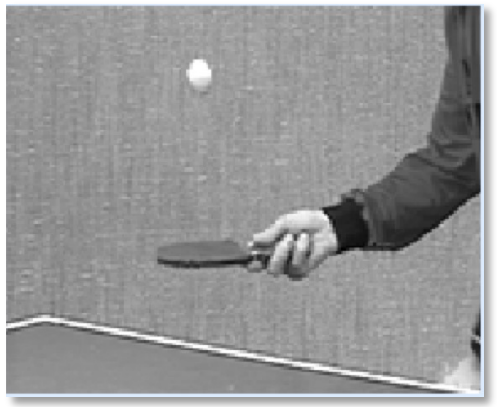
$$I^{MC}(\vec{r},t) = I(\vec{r} - \hat{\vec{D}}(\vec{r}), t - \Delta t)$$



$I(\vec{r},t-\Delta t)$



$\hat{\vec{D}}(\vec{r})$

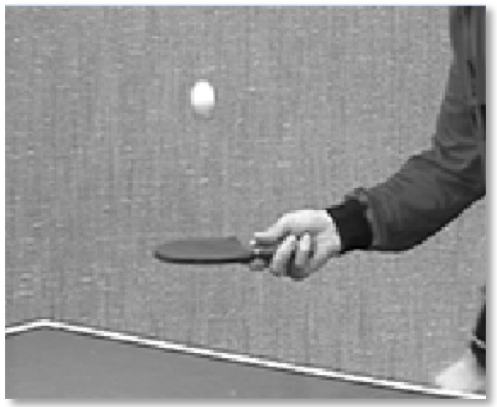


$I^{MC}(\vec{r},t)$

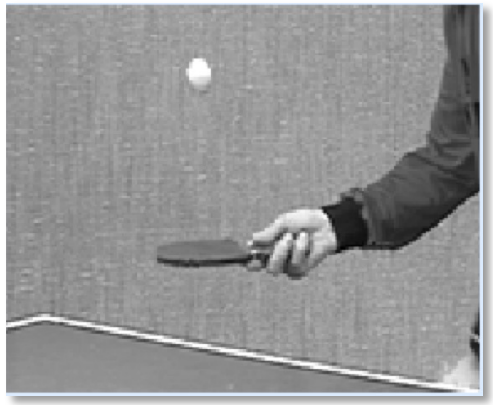
# 2D Motion analysis

The difference between the motion compensated frame and the real current image is called the **Displaced Frame Difference** (DFD).

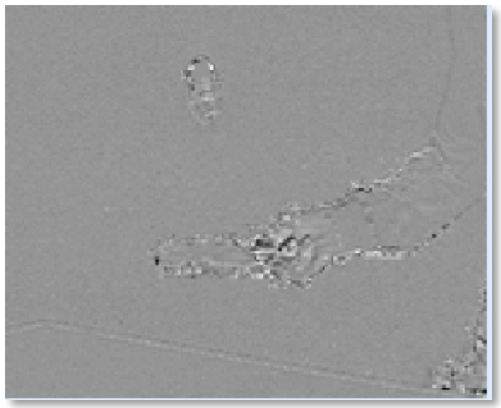
$$DFD(\vec{r},\hat{\vec{D}}(\vec{r}))=I(\vec{r},t)-I(\vec{r}-\hat{\vec{D}}(\vec{r}),t-\Delta t)$$



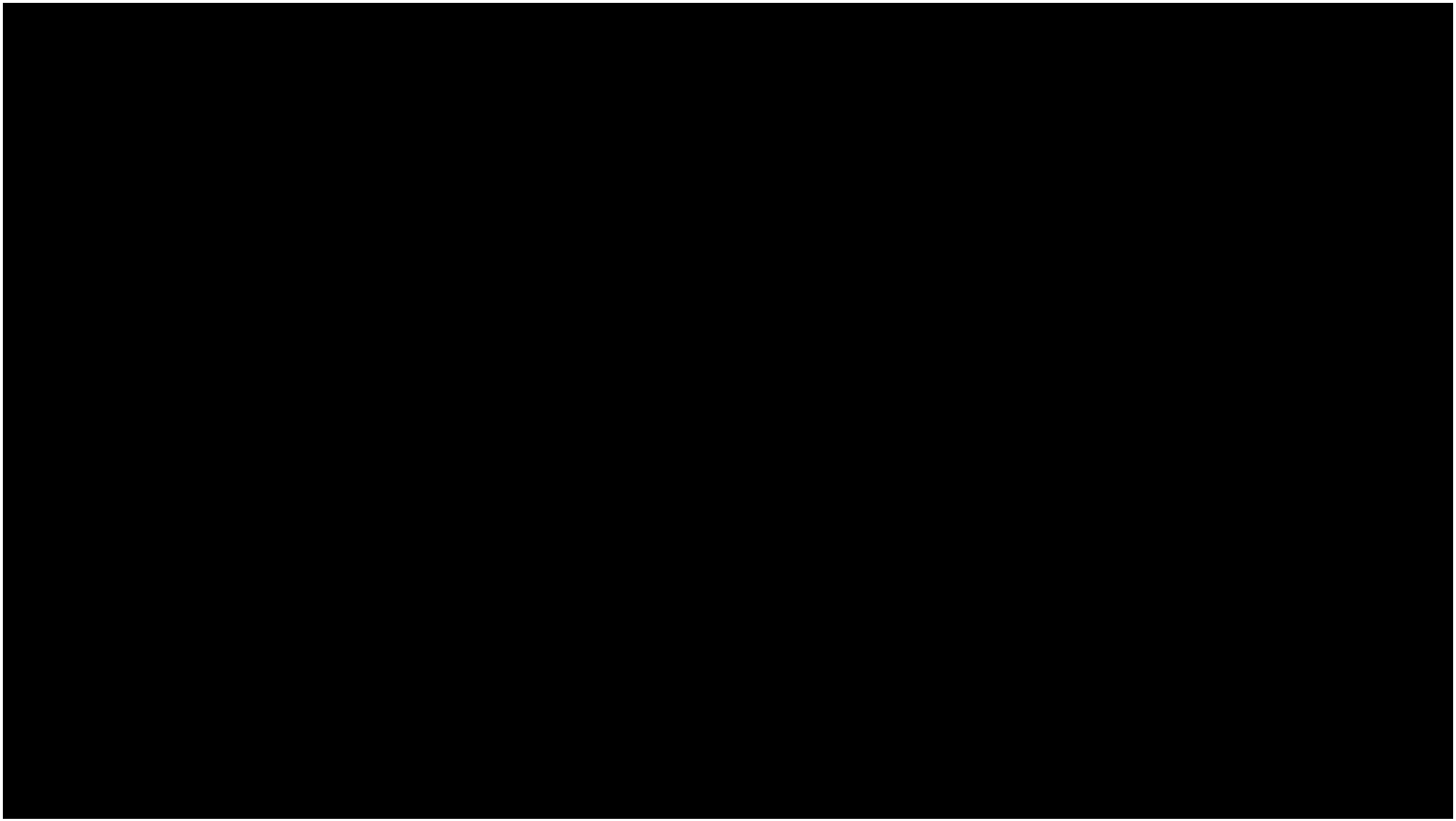
$I(\vec{r},t-\Delta t)$



$I^{MC}(\vec{r},t)$

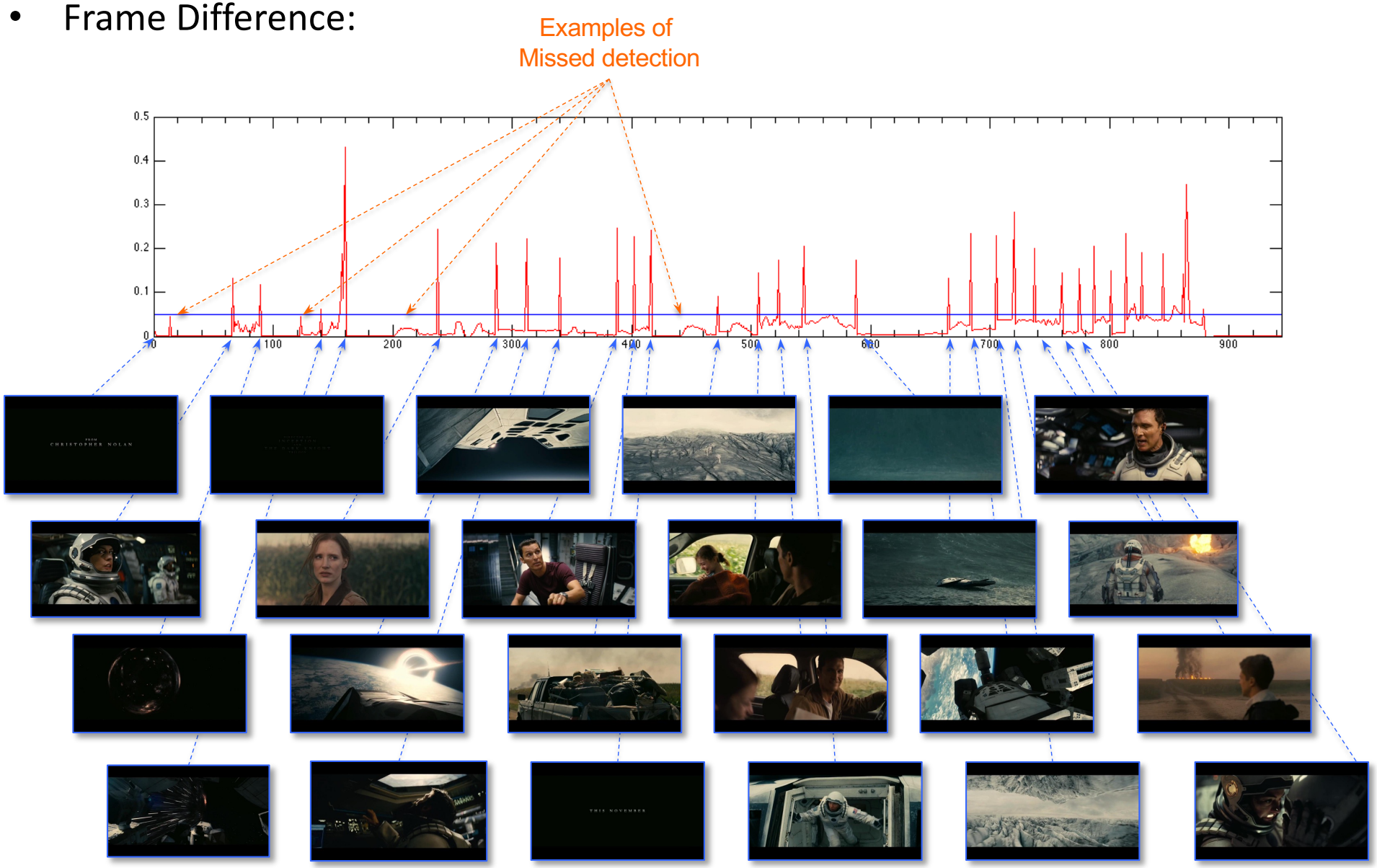


$DFD(\vec{r},\hat{\vec{D}}(\vec{r}))$

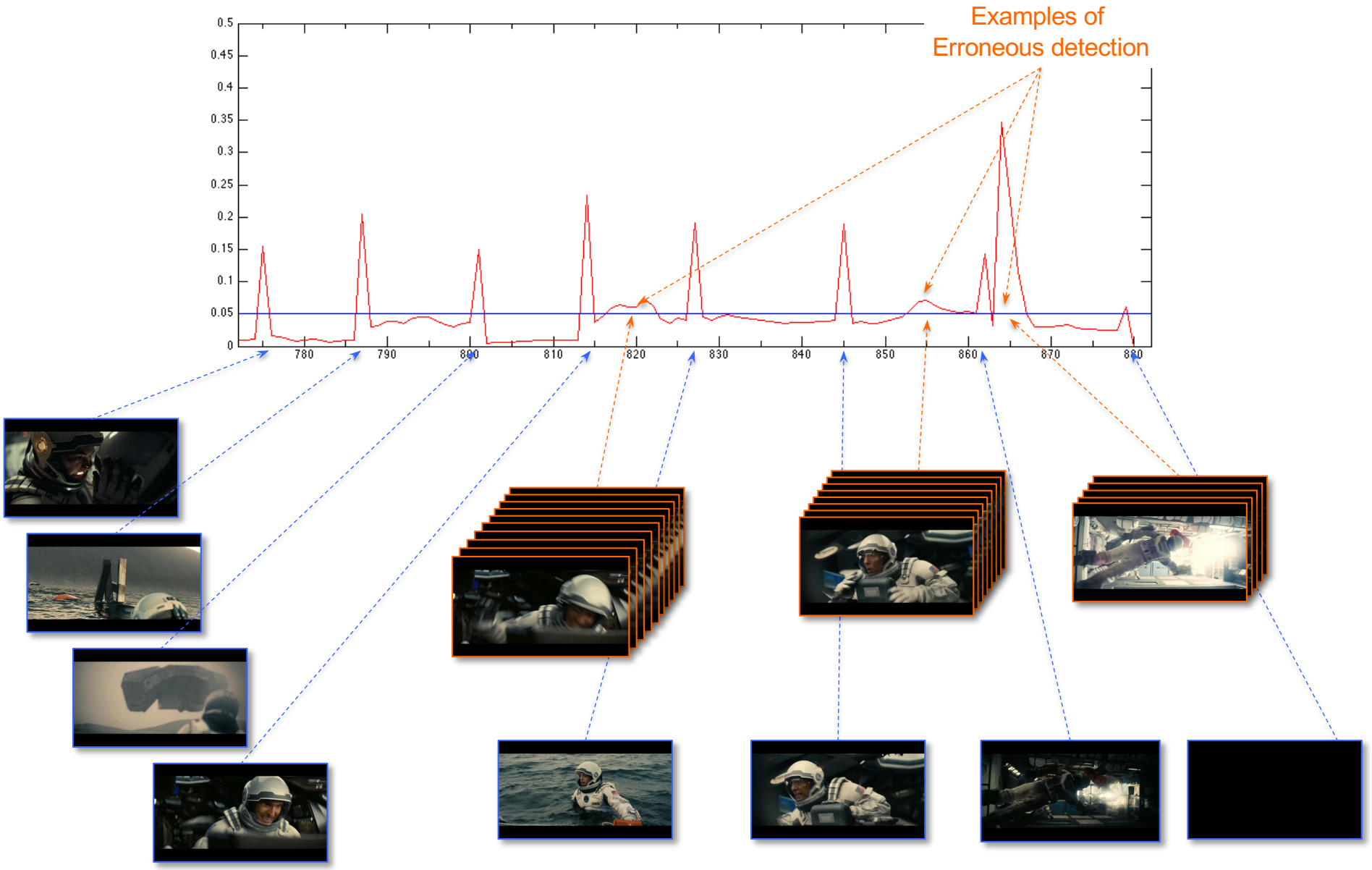


# Shot segmentation (II)

- Frame Difference:

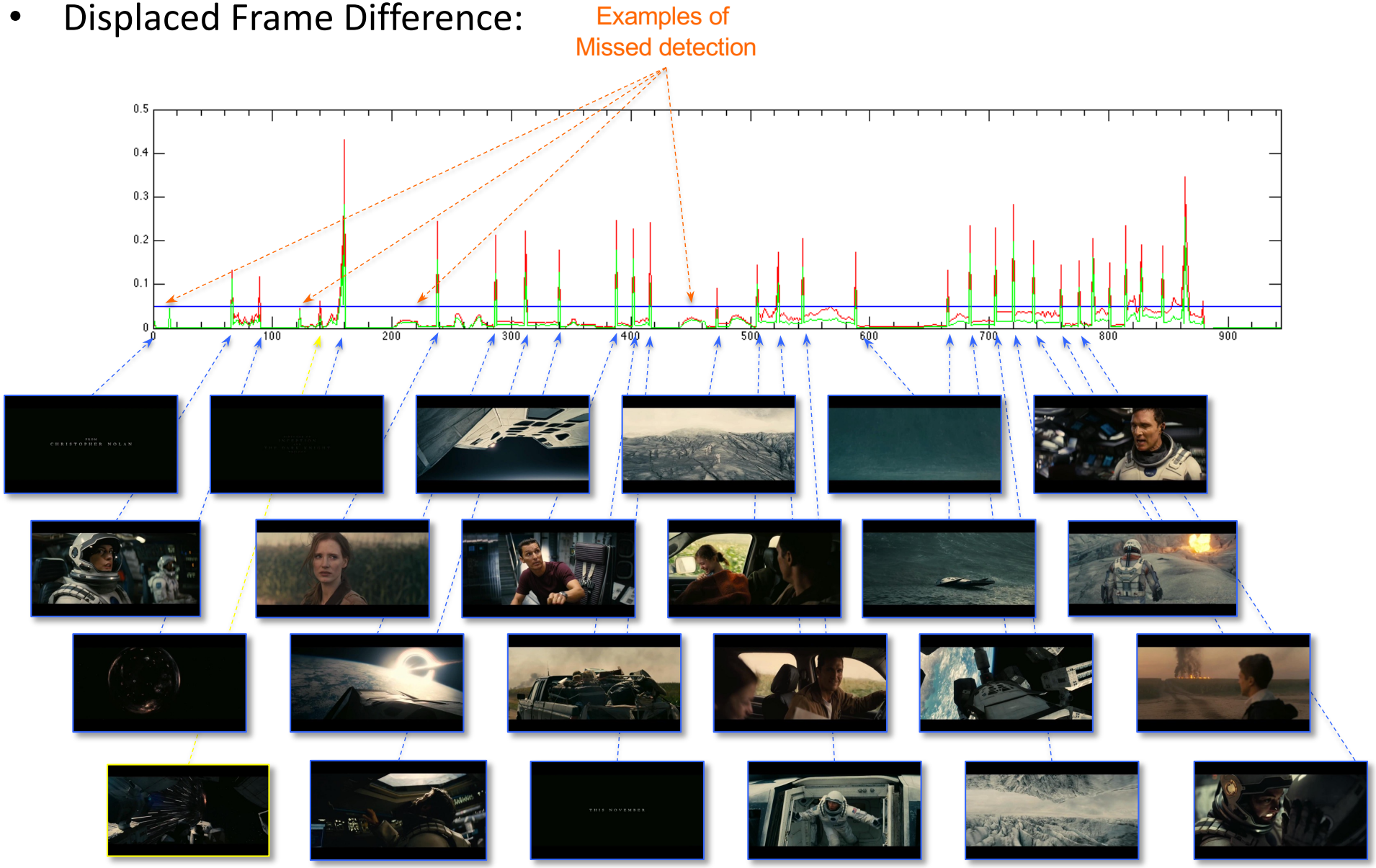


# Shot segmentation (III)

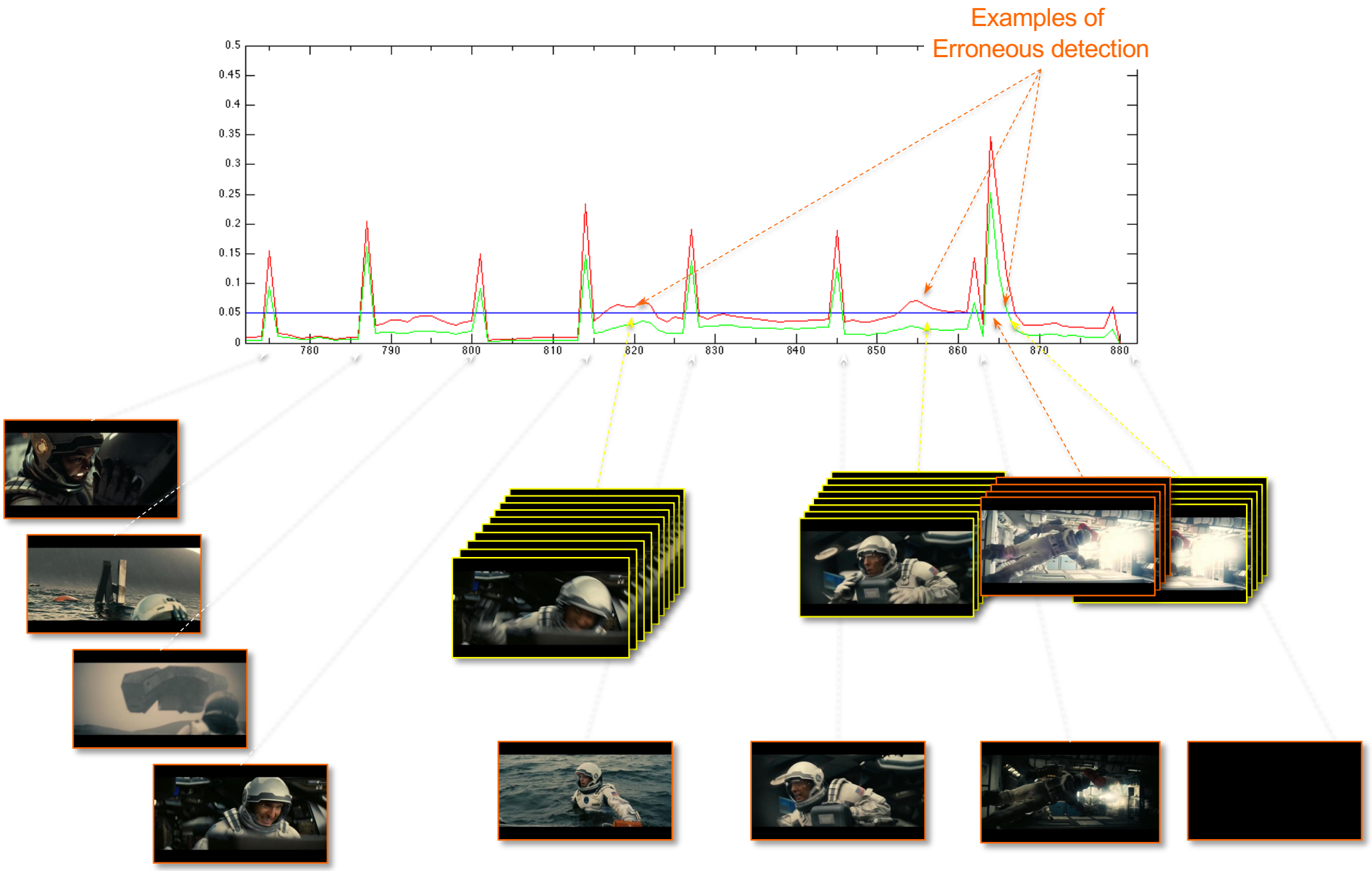


# Shot segmentation (IV)

- Displaced Frame Difference:



# Shot segmentation (V)



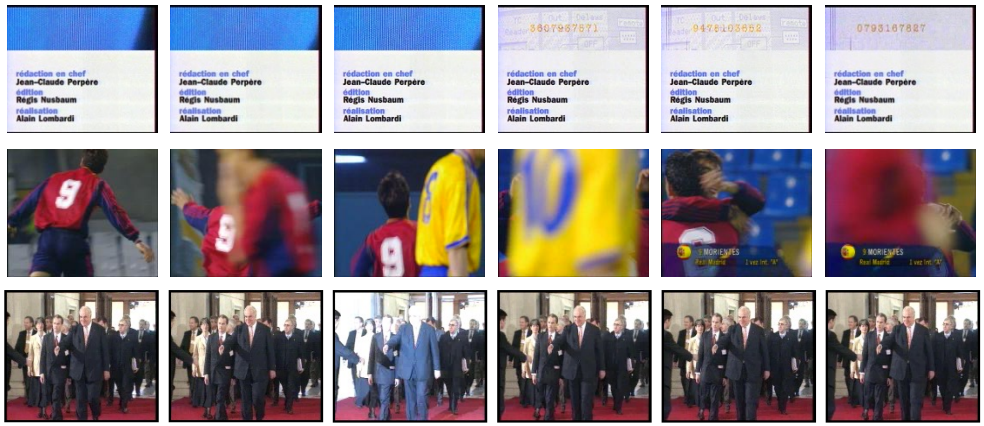
# Shot segmentation (VI)

The DFD comparison allows predicting the current image with the information of the previous image leading to a more robust estimation of the shot transition. However, some transitions are difficult to handle (dissolve and fade, for instance)

- Examples of **subtle shot transitions**.

**These changes correspond to shot transitions**

- Examples of **internal transitions or motion**.



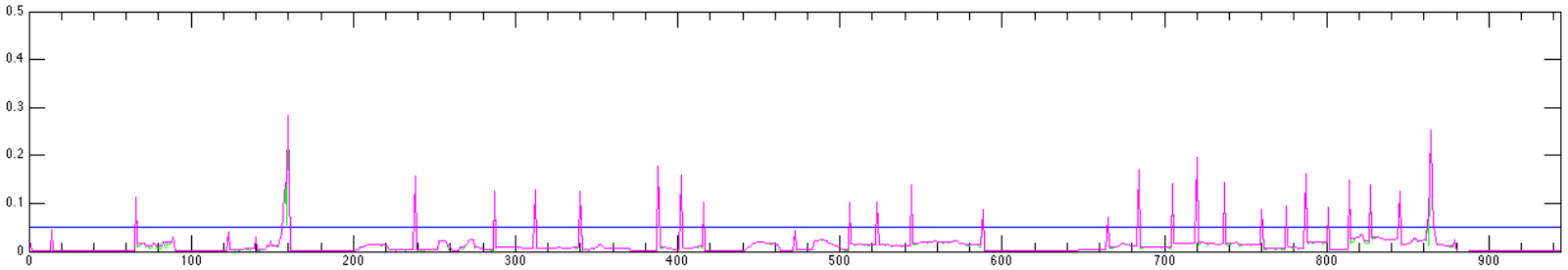
**These changes *do not* correspond to shot transitions**

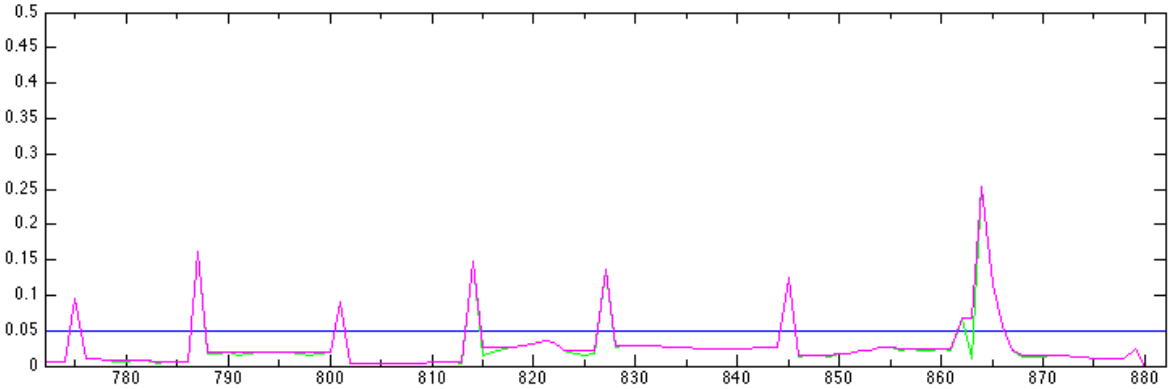
# Shot segmentation (VII)

- Direct binarization may not be very robust in practice -> **Segmentation pb**

- **Simplification**:
  - Remove minima that are not long enough to correspond to a shot: closing with a structuring element corresponding to the shortest expected shot:

$$f(t) = \varphi_w(DFD(t))$$

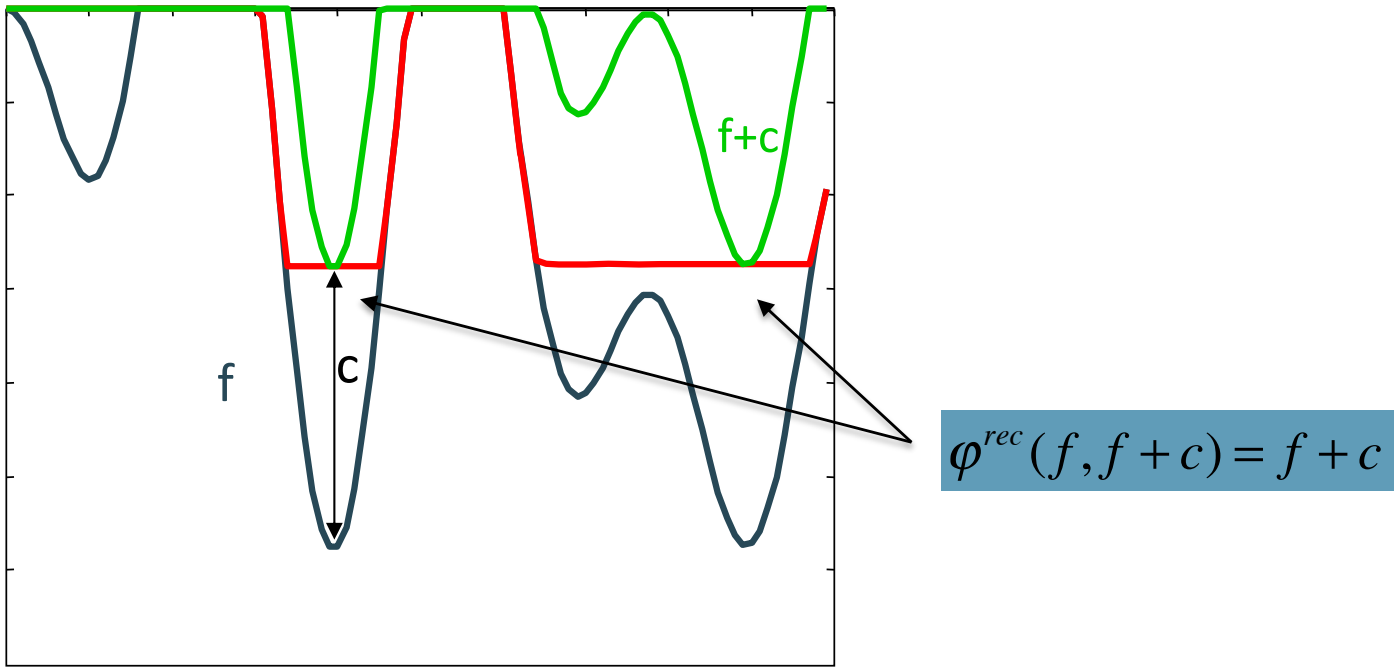


# Shot segmentation (VIII)

## Basic segmentation steps

- **Decision**:
  - Instead of binarization, marker detection and then watershed (or region growing)
  - Selection of meaningful minima: Negative contrast filter

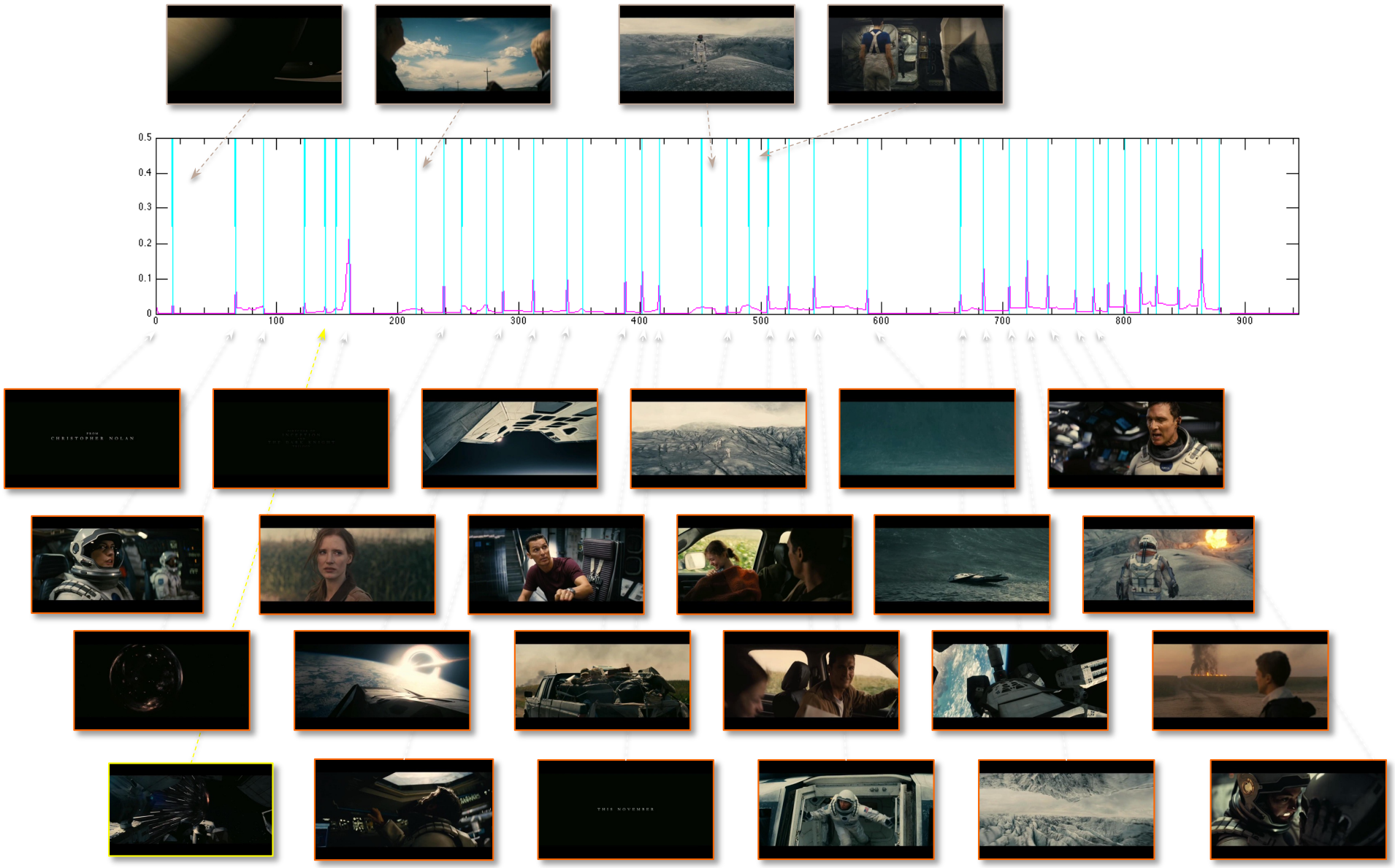


# Shot segmentation (IX)

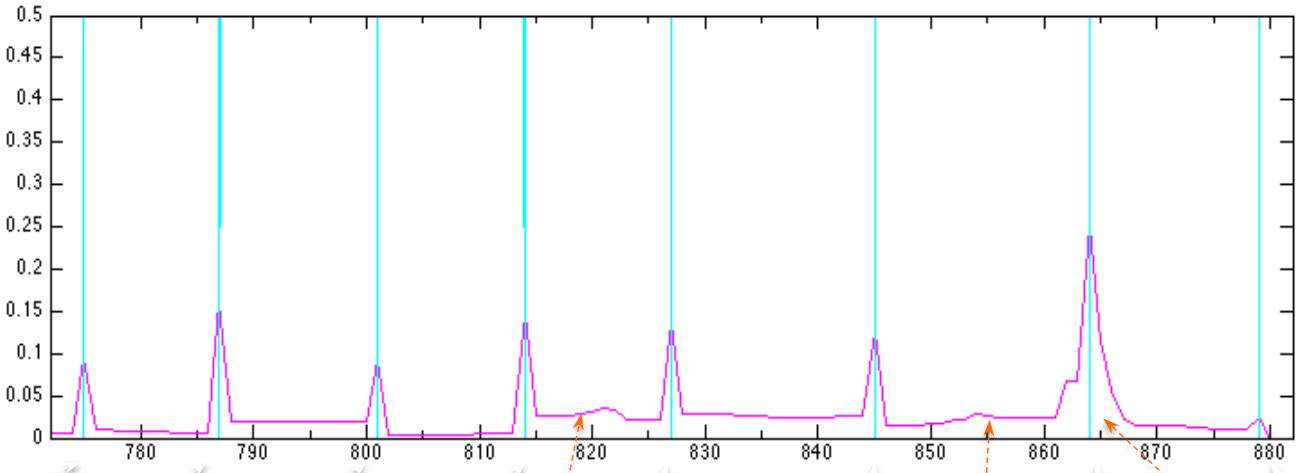
## Temporal segmentation: Decision

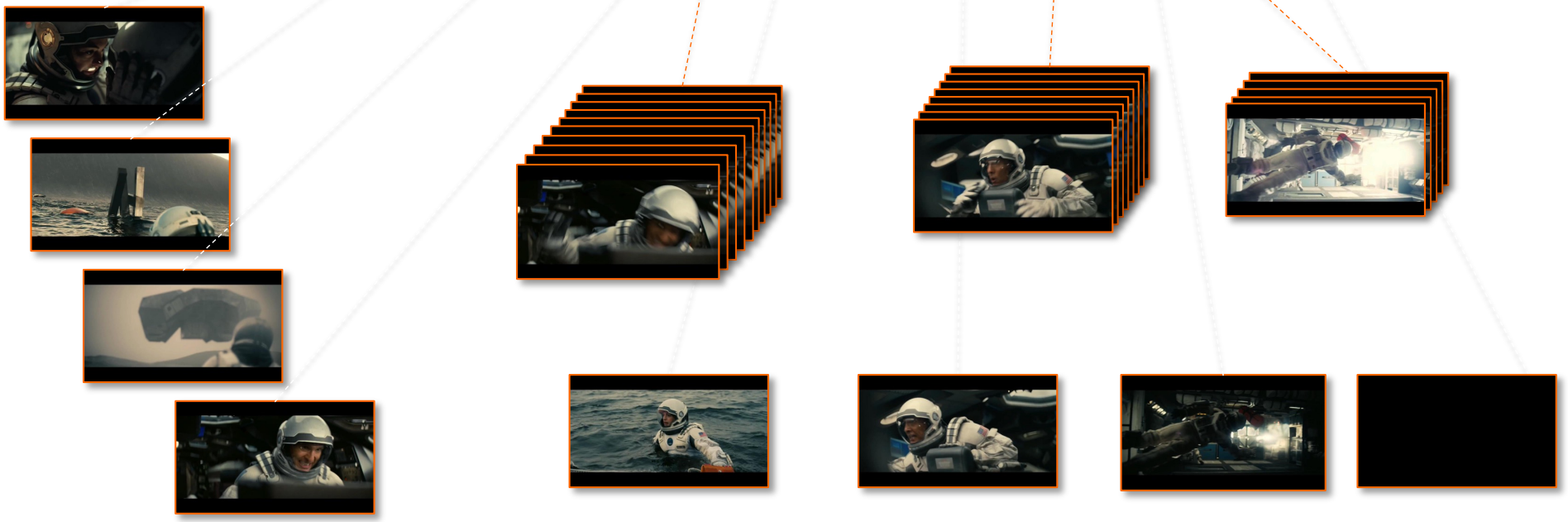
- **Watershed**: Flooding start by the markers

# Shot segmentation (X)



# Shot segmentation (XI)



# Shot segmentation (XIII)

- End-to-end Fully Convolutional Neural Network

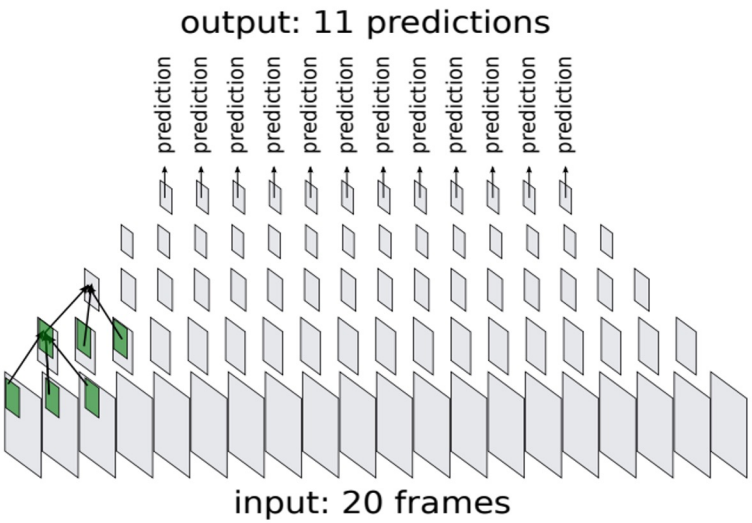



Figure 2: Our network architecture. Each frame-prediction is based on a context of 10 frames. By using a model that is fully convolutional in time, we can increase the input size and thus make *e.g.* 11 predictions by analyzing 20 frames or 91 predictions by analyzing 100 frames, etc., thus minimizing redundant computation.

M. Gigly, Ridiculously Fast Shot Boundary Detection with Fully Convolutional Neural Networks. CBMI 2018

# Outline

- Introduction to video segmentation
- Shot segmentation
- Moving object segmentation
    - Introduction
    - Frame differencing
    - Still background estimation
    - Variable background estimation: Single Gaussian
    - Variable background estimation: Multiple Gaussians
    - Other change detection techniques
  - Shadow detection
  - Connected component analysis and tracking
- Region segmentation

# Moving object segmentation

- Detect and segment moving objects from a video sequence of a fixed camera
  - Background – static scene
  - Foreground – moving objects



# Moving object segmentation: Introduction

- Also called Change detection and Background subtraction
- Some applications do not require the estimation of the motion in the scene, but only the knowledge of whether any object(s) in the scene has(ve) moved and an approximate position.

***Security systems***

***Selective coding techniques***

# Moving object segmentation: Introduction

- The technique can rely on a **pixel-based model** of the image:
  - Classify each pixel as belonging to the static (**background)** or moving object (**foreground**) class

- Connected component analysis and tracking are used for further analysis

# Moving object segmentation

- Compute the **pixel to pixel difference** between consecutive images: **frame differencing or change detection**
  - Subtraction of images (frame differencing)
  - Thresholding

- Compute the **pixel to pixel difference** between an image and an estimated background image: **background subtraction**
  - Initialize background model
  - Subtraction and classification(compare current frame with the model)
  - Update the model

# Frame differencing

$$\left|I(x,y,t) - I(x,y,t-1)\right| > Th$$

Image at time t: I(x,y,t)


Image at time t-1: I(x,y,t-1)


- Depending on the object structure, speed, frame rate and global threshold, this approach may or may not be useful.
- Difficult to establish the threshold
- No problem with illumination changes
- May obtain just the silhouette for large homogeneous objects

Slide credit: Birgi Tamersoy

# Frame differencing

$Th = 25$

$Th = 50$

$Th = 100$

$Th = 200$

Slide credit: Birgi Tamersoy

# Moving object detection

- A reliable and robust moving object detection algorithm should handle:
  - Sudden or gradual illumination changes,
  - High frequency, repetitive motion in the background ( tree leaves, waves, …)
  - Long-term scene changes (a car is parked for a long period).

- Background subtraction: Compute the **pixel to pixel difference** between an image and an **estimated background image**
  - Initialize background model
  - Subtraction and classification(compare current frame with the model)
  - Update the model

# Still background estimation: Introduction

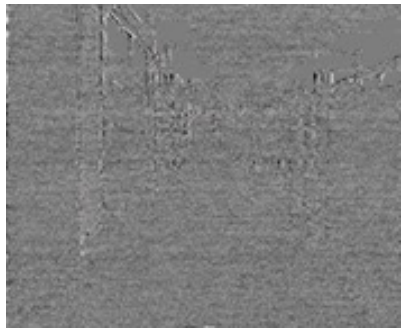
- In several security applications, due to **the setting** and **camera configuration**, change detection can be understood as a problem of **still background estimation**.



- A static camera is observing a scene (**background**) that, in principle, **does not vary**. However, variations appear: camera sensors introduce **noise**.
- A **variation** in the scene (something appearing in the **foreground**) can be detected by comparison with an estimation of the still background.

# Still background estimation: Signal model (I)

- Every pixel of the **background** is modeled as a **random variable**:
  - Its **mean** ($\mu$) represents its actual value
  - Its **variation** with respect to the mean value is due to the noise introduced by the camera sensor.
    - Camera sensor pixels are usually assumed to be **independent** and **similar**. Therefore, the noise variables at every pixel can be modeled as independent and identical distributed (**iid**) random variables.
    - The noise probability distribution is commonly modeled as **Gaussian** $N(0, \sigma^2)$



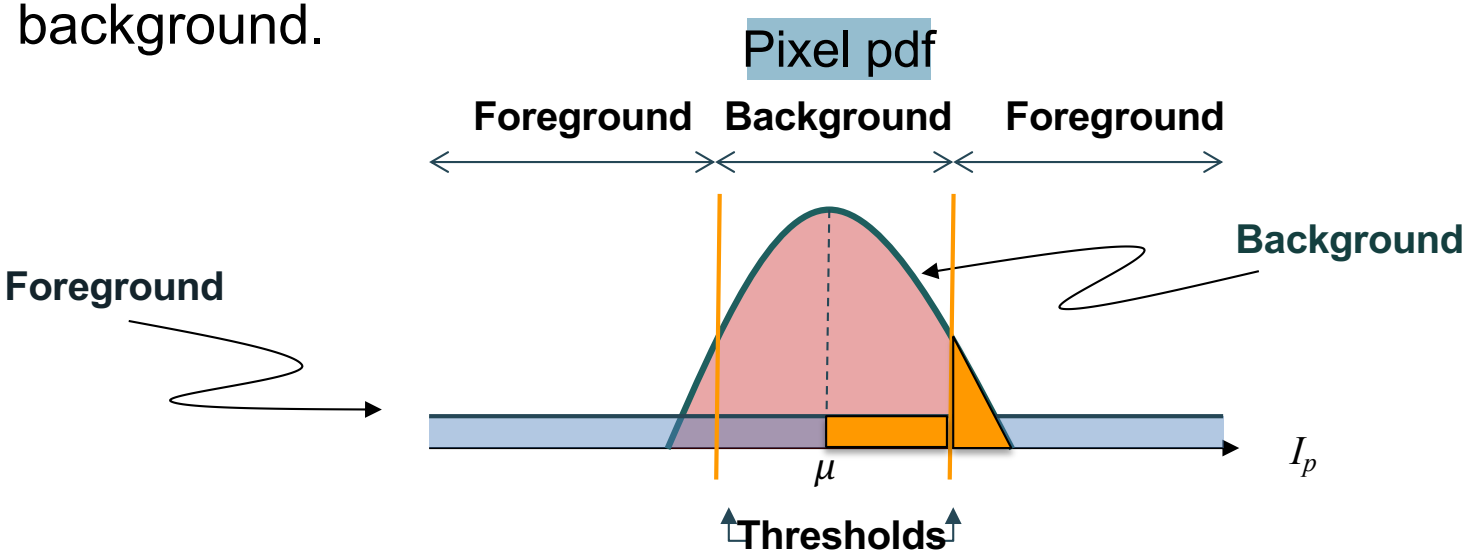
**Magnified Error image**

# Still background estimation: Signal model (II)

- Every pixel of the **foreground** is modeled as a **random variable**:
  - The foreground is difficult to model since **no a priori information** can be assumed.
  - The **source of information** can be related to intruders or even to artifacts produced in the recording of the scene.
    - A **uniformly distributed probability function** is commonly assumed

# Still background estimation: Classification

- Once background and foreground have been statistically modeled, changes in the scene can be detected by means of a **classification process**.
- A **Maximum Likelihood** classifier is commonly used, that minimizes the probability of error in the classification
- Every pixel ($p$) in a new image is **separately analyzed**. If the pixel value falls in-between the two thresholds, the pixel is classified as background.



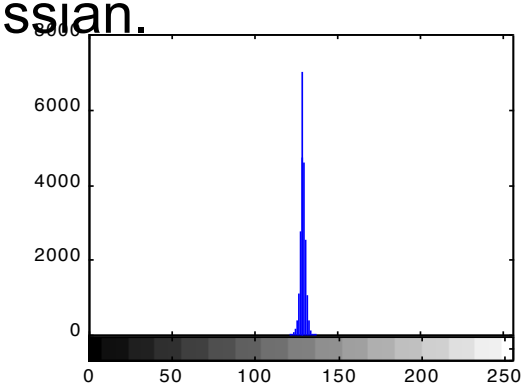
# Still background estimation (I)

- Various realizations of the background image are used to model each pixel

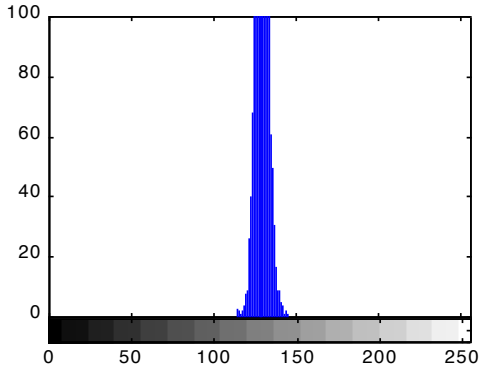
**Realizations of the *Background* process**

# Still background estimation (II)

- The **mean values** of the pixels are estimated by **averaging** the various realizations.
- The Frame Difference **histograms** give an estimate of the probability density functions of the noise.
  - Note that here the histogram **is not correctly modeled** by a Gaussian.




**Histogram**




**Zoom of the histogram**

Frame Difference is displaced to 128 in order to have positive values

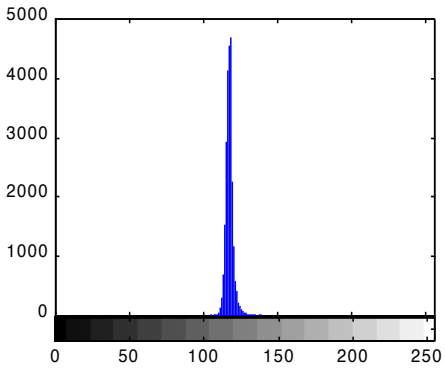
Thresholds are fixed at pixel values 108 and 148 (**mean ± 20**)

# Background/Foreground classification example (I)

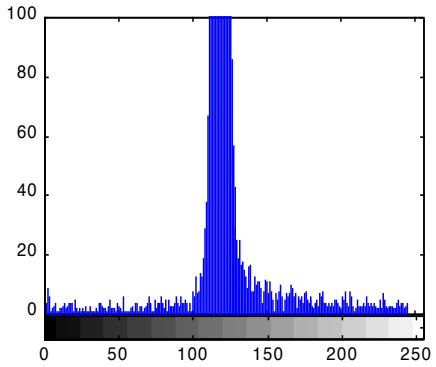
**Background image**

**Input image**

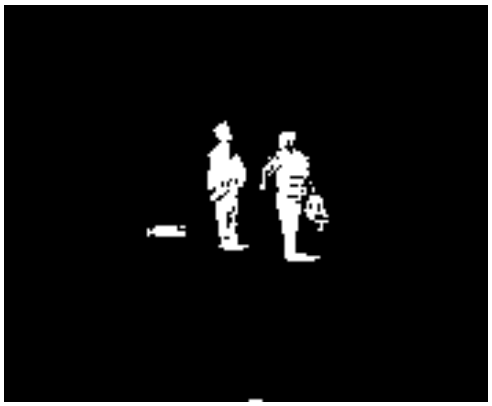
**Difference image**




**Pixel Histogram**




**Zoom of the histogram**



**Background subtraction result**
**Thresholds: mean ± 20**

# Background/Foreground classification example (II)

**Segment of the original sequence**
***Hall Monitor***

# Background/Foreground classification example (II)

**Change detection on *Hall Monitor***

# Problems of still background estimation

- The **variations in the background pixel values** can be larger than expected because of:
  - **Gradual changes in the illumination** of the scene
  - **Sudden changes in the illumination** of the scene
  - **Exterior scenes**: typically, this leads to non-static backgrounds.
  - ✓ Techniques for **variable background estimation** are commonly used.

# Outline

- Introduction to video segmentation
- Shot segmentation
- Moving object segmentation
    - Introduction
    - Frame differencing
    - Still background estimation
    - **Variable background estimation: Single Gaussian**
    - **Variable background estimation: Multiple Gaussians**
    - **Other change detection techniques**
  - **Shadow detection**
  - **Connected component analysis and tracking**
- **Region segmentation**
    - **Spatial segmentation and tracking**
    - **Spatio-temporal segmentation**

# Variable background estimation: Introduction

- In some security applications, due to **the setting** and **camera configuration**, the change detection can be understood as a problem of **variable background estimation**.


- A static camera is observing a scene (**background**) that **may slowly vary** (day light changes, clouds, etc). In addition, smaller variations appear due to camera sensor **noise**.
- A **variation in the scene** (something appearing in the **foreground**) can be detected by comparison with an estimation of the **current background**.

# Variable background estimation: Single Gaussian (I)

- Every pixel of the **background** is modeled as a **random variable**:
  - Its mean ($\mu_p$) may suffer **slow changes through time** (which depend on the illumination at each moment)

  - **Variations** with respect to the mean are due to the noise introduced by the camera
    - Noise samples are assumed to be **independent** and modeled as **Gaussian** $N(0, \sigma_p^2)$ functions.

# Variable background estimation: Single Gaussian (II)

- Background mean and variance are estimated as the mean of n frames values:

$$\mu_p(t) = \frac{1}{n}\sum_{i=0}^{n-1} I_p(t-i) \qquad \sigma_p^2(t) = \frac{1}{n}\sum_{i=0}^{n-1}\left(I_p(t-i) - \mu_p(t)\right)^2$$

- High memory requirements -> Running average

$$\mu_p(t) = (1-\rho)\mu_p(t-1) + \rho I_p(t)$$

The parameter $\rho$ establishes the **memory of the system**:
$\rho$ -> *0* implies no updating (**long memory**)

- Background mean estimated as the median of the previous n frames values:

$$\mu_p(t) = \underset{i\in\{0,\ldots,n-1\}}{median}\left\{I_p(t-i)\right\}$$

- Accuracy depends on object speed and frame rate

# Variable background estimation: Classification (I)

- Once the background has been modeled, changes in the scene can be detected by means of a **classification process**.
- As before, a **Maximum Likelihood** classifier minimizing the classification error can be used. But it is usually treated as a one-class classification problem: thresholds are fixed around the means.
- Typically, a Gaussian model is used and thresholds = $\mu \pm 2.5\sigma$.
- Every pixel ($p$) in a new image is **separately analyzed**. If its value is within the range given by the thresholds, $I_p \in [\mu - 2.5\sigma, \mu + 2.5\sigma]$, the pixel is classified as background and no changes in the scene are said to be detected at this pixel position.

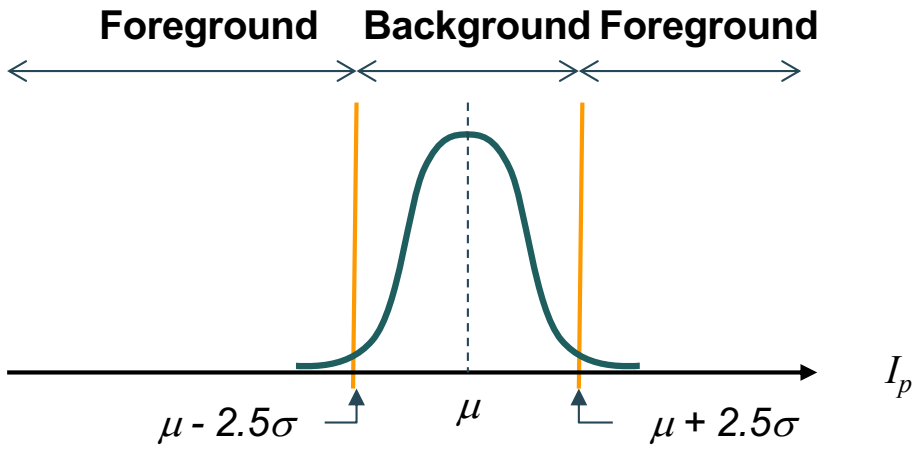
# Variable background estimation: Classification (II)

- The background is modeled by a set of **NxM random Gaussian variables** ($I_p$):

$$f(I_p) = \frac{1}{\sqrt{2\pi\sigma_p^2}} \exp\left( -\frac{(I_p - \mu_p)^2}{2\sigma_p^2} \right)$$

- A pixel ($p$) is assigned to the **background** if:

$$|I_p - \mu_p| < 2.5\sigma_p$$



# Mean and median filter

- For $n = 20$:

Estimated Background

Foreground Mask

Mean


- For $n = 20$:

Estimated Background

Foreground Mask

Median


Slide credit: Birgi Tamersoy

# Variable background estimation: Classification (III)

- In the case of **color images**, the background is modeled by a set of **NxM 3D random Gaussian variables** ($\underline{I}_p$):

$$f(\underline{I}_p) = \frac{1}{\sqrt{(2\pi)^3 \left|\underline{\underline{\Sigma}}_p\right|}} \exp\left( -\frac{1}{2} [\underline{I}_p - \underline{\mu}_p]^T \underline{\underline{\Sigma}}_p^{-1} [\underline{I}_p - \underline{\mu}_p] \right)$$

- Commonly, the three components are assumed to be independent:

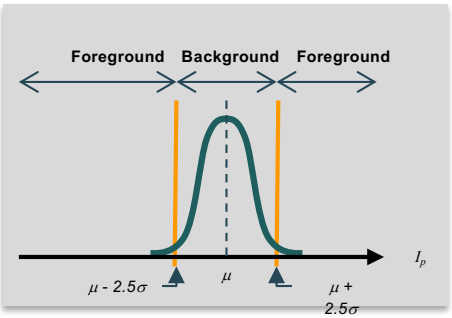
$$f(I_p) = \prod_{d=1}^{3} \frac{1}{\sqrt{2\pi \left[\sigma_p^2\right]_d}} \exp\left( -\frac{1}{2\left[\sigma_p^2\right]_d} \left( \left[I_p\right]_d - \left[\mu_p\right]_d \right)^2 \right)$$

# Variable background estimation: Running Gaussian Average

- The process is divided into two steps:
- **Initialization**: During a training period, the initial **mean** and **variance** values of every Gaussian variable $(\mu_p, \sigma_p^2)$ are estimated.
- **Updating**: Mean and variance of every Gaussian variable are updated based on **incoming pixels classified as background**

$$\mu_p(t) = \begin{cases} (1-\rho)\mu_p(t-1) + \rho I_p(t) & if\ p \in Background \\ \mu_p(t-1) & if\ p \in Foreground \end{cases}$$

$$\sigma_p^2(t) = \begin{cases} (1-\rho)\sigma_p^2(t-1) + \rho\left(I_p(t) - \mu_p(t)\right)^2 & if\ p \in Background \\ \sigma_p^2(t-1) & if\ p \in Foreground \end{cases}$$



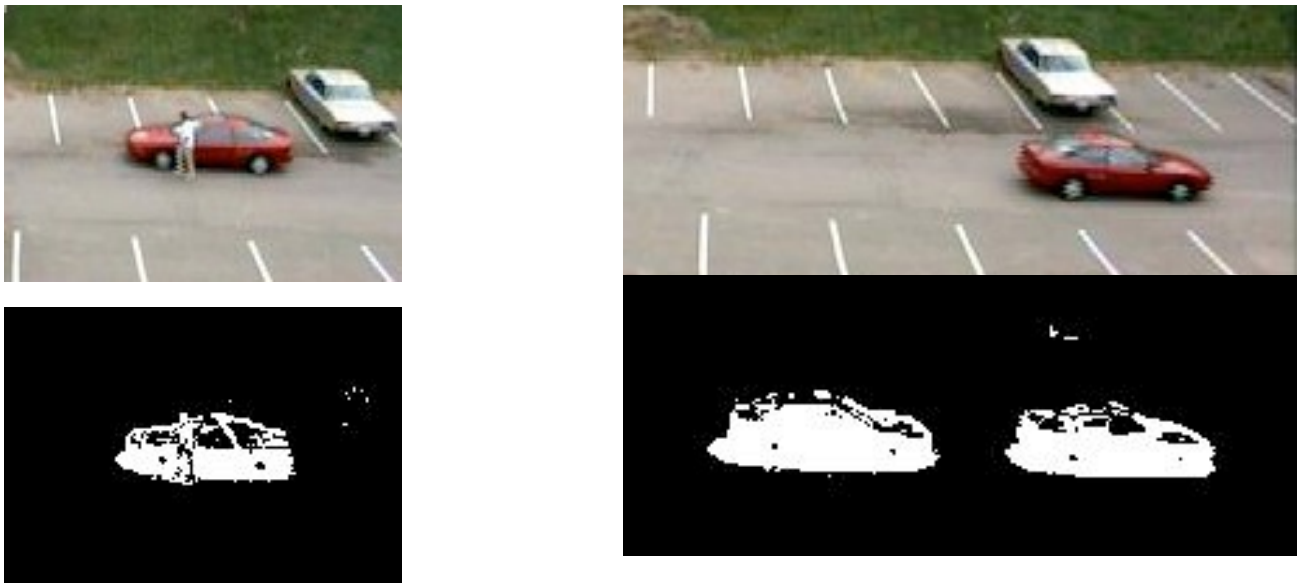
# Variable background estimation: Introduction (I)

- In several security applications, systems have to model **large variability of the background** due to switch between **various states of the background**

- A static camera is observing a scene (**background**) on which several pixels **may vary their values due to different time instances of the background scene**; for instance, a pixel may represent a leaf of a tree or, due to the wind, the building behind the tree. In addition, **smaller variations** appear due to camera sensor **noise**.

# Variable background estimation: Introduction (II)

- Moreover, in several security applications, **new objects** that remain in the scene for a long period have to **be assimilated to the background.**



- This **variation has to be tracked** and its duration monitored to be able to **include it in the background** after some time (e.g.: a car that remains parked or leaves the parking place)

# Variable background estimation: Multiple Gaussians (I)

- Every pixel of the **background** is modeled as a **random variable**:
  - **Random variations** with respect to its mean value are due to two different sources:
    - The **noise** introduced by the camera sensors.
    - The **variability of the scene**: different background objects may be observed at the same location. The mean suddenly changes: **Switch between various states.**

- This leads to a statistical model of the background based on multiple Gaussians: **Gaussian Mixture Model** (GMM)

# Variable background estimation: Multiple Gaussians (II)

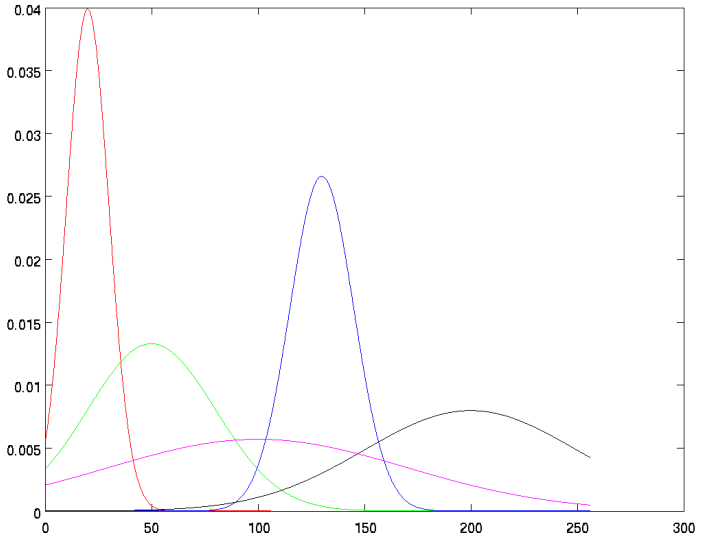
- The Stauffer & Grimson approach also uses **Gaussians to model foreground objects**:
  - We have no a priori information about foreground objects….. But once they have appeared, they can be modeled (as their appearance is generally stable).
  - The use of Gaussian models for foreground objects allows us to easily **integrate them in the background** after some time if they remain still.

- Both models (background and foreground) are mixed into a **common model for the pixel observation**, the **Gaussian Mixture Model** (GMM)

# Variable background estimation: Multiple Gaussians (III)

- At each pixel:
  - Different Gaussians model **different color appearances** of the scene
  - Each Gaussian represents **either a foreground object** or **some time instances of the background** scene.

$$f(I_p(t)) = \sum_{i=1}^{K} w_i(t) N\left(\mu_i(t), \sigma_i(t)\right)$$



# Variable background estimation with GMM (I)

- The estimation process is based on the temporal observation of the pixel $I_p(t)$, t=0,…,n
  - The analysis memory is a **design parameter**.
  - Short memory allows to adapt rapidly to scene changes (but may not be very robust). Long memory is more robust (but may have difficulties in dealing with normal background variation)

- The estimation assumes that the pixel observations can be correctly modeled by means of $K$ **Gaussians**:
  - The number of Gaussian functions ($K$) **adapts to the data**

- The parameters of the Gaussian functions ($w_i$, $\mu_i$, $\sigma_i$ ) are estimated at **every time instant** ($t$) (**adaptive** gaussian mixture).

# Variable background estimation with GMM (II)

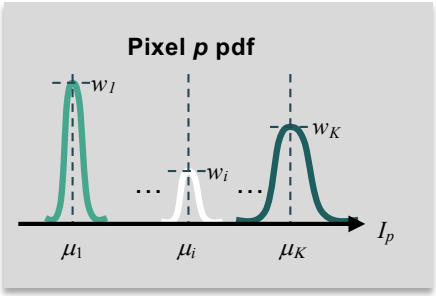
- Incoming pixels are either classified into one of the **existing Gaussians or a new Gaussian** is created. In this case the existing Gaussian with less probability (w/$\sigma$) is eliminated.
  - The **previous classification approach** is commonly used:

$$\left|I_p(t)-\mu_{p,i}(t-1)\right|<2.5\sigma_{p,i}(t-1) \quad \Rightarrow \quad p(t)\in N_i$$

- Means and variances of Gaussians are adapted **based on the classified incoming pixels**:
  - They can be computed using the **previous estimation approach**

$$\mu_{p,i}(t)=\begin{cases}(1-\rho)\mu_{p,i}(t-1)+\rho I_p(t) & if\ p(t)\in N_i\\ \mu_{p,i}(t-1) & if\ p(t)\notin N_i\end{cases}$$

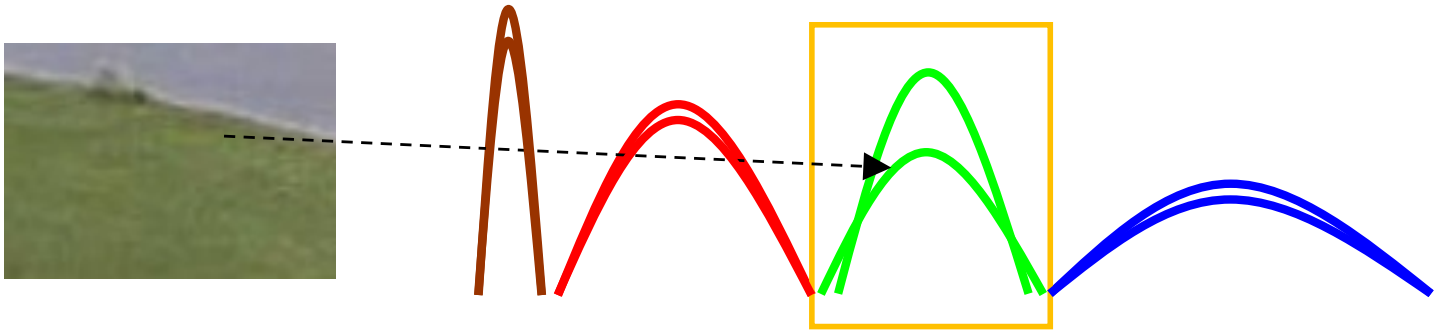
$$\sigma^2_{p,i}(t)=\begin{cases}(1-\rho)\sigma^2_{p,i}(t-1)+\rho\left(I_p(t)-\mu_{p,i}(t)\right)^2 & if\ p(t)\in N_i\\ \sigma^2_{p,i}(t-1) & if\ p(t)\notin N_i\end{cases}$$



# Variable background estimation with GMM (III)

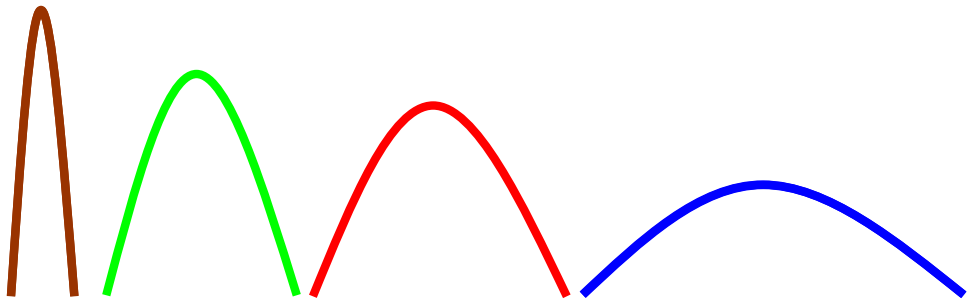
- Gaussians are weighted depending on the **frequency of the decisions**
  - The Gaussian to which the current pixel is assigned **increases** its weight
  - The weights of the remaining Gaussians **decrease**

$$w_{p,i}(t) = \begin{cases} (1-\alpha)w_{p,i}(t-1)+\alpha & if\ p(t) \in N_i \\ (1-\alpha)w_{p,i}(t-1) & if\ p(t) \notin N_i \end{cases}$$



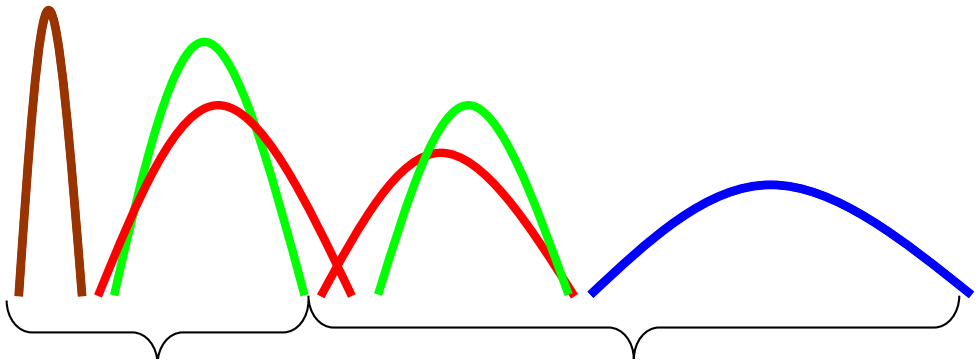
# Variable background estimation: Classification (I)

- Which Gaussians correspond to background or foreground objects?
- Background pixels are modeled by Gaussians with **high weight** and **low variances**:
  - **High weight**: Background pixels appear **many times**
  - **Low variance**: Background pixels are always **very similar**
- Gaussians are **reordered** into descending $w/\sigma$



# Variable background estimation: Classification (II)

- The first *B* Gaussians that **represent a fair % of observations** (represented by the threshold *T*) are said to model the Background



$$B = \underset{K}{\operatorname{argmin}}\left(\sum_{j=1}^{K} w_j > T\right)$$

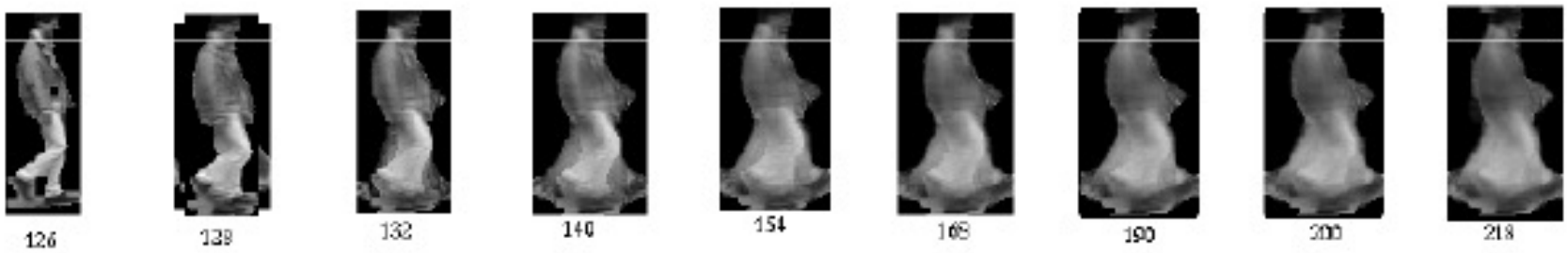
- A pixel in a given time instant ($p(t)$) is **classified as Background** if the Gaussian representing it is among these first B Gaussians.

# Variable background estimation with GMM: Examples

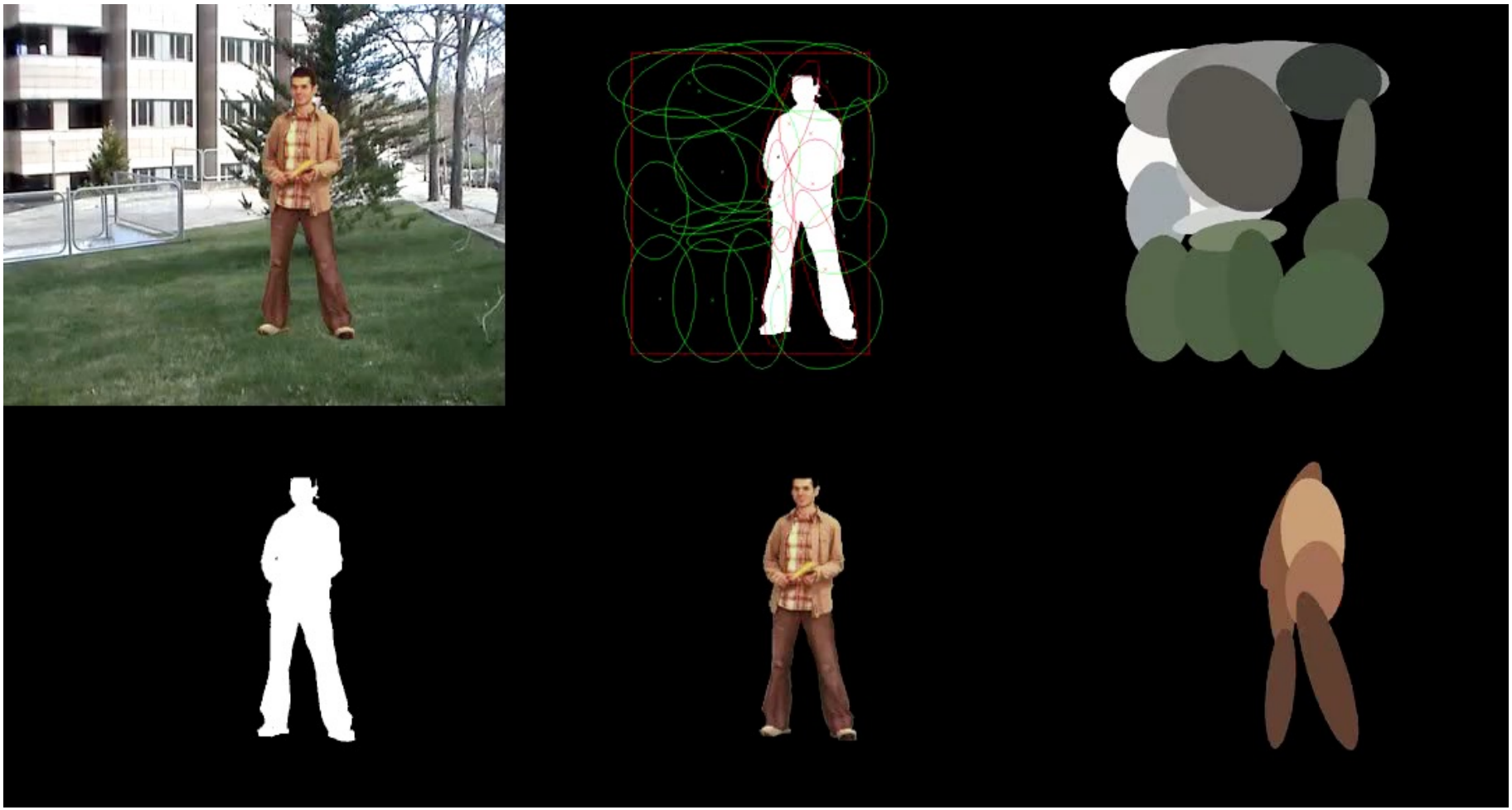
# Foreground and background model

- *W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People, I. Haritaoglu, D. Harwood and L. S. Davis, International Conference on Face and Gesture Recognition, April 14-16, 1998*

  - Models the object to improve localization and tracking through occlusions
  - Constructs a texture template that is updated along time



# Foreground and background models



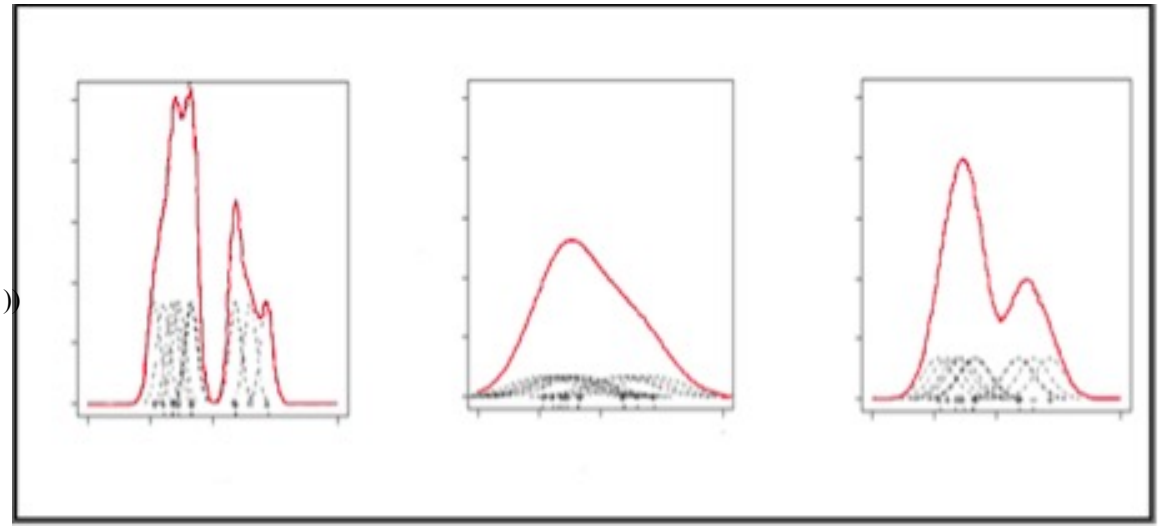
# KDE: Kernel Density Estimation

Elgammal et al:

- The background $PDF_j$ at pixel j is given by the histogram of the *N* most recent pixel values, each smoothed with a Gaussian kernel (sample-point density estimator). If $PDF_j(x_t) > Th$, the j pixel is classified as background
- It is able to adapt very quickly to changes in the background process and to detect targets with high sensitivity.
- Problems: memory requirement (n * size(frame)), time to compute the kernel values (mitigated by a LUT approach)

$$f(I_p)\Big|_t = \frac{1}{n}\sum_{i=1}^{n} K(I_p - I_p(i))$$

$$f(I_p)\Big|_t = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(I_p - I_p(i))^T \Sigma^{-1}(I_p - I_p(i))}$$



# Background subtraction. Eigenbackgrounds

N. Oliver et al:

N images are used to construct the background model. Eigenvector decomposition is computed and dimensionality reduction is achieved with Principal Component Analysis (PCA)

PCA is applied to a sequence of N frames to compute the *eigenbackgrounds*

Input images are projected on the space expanded by the eigenbackground images

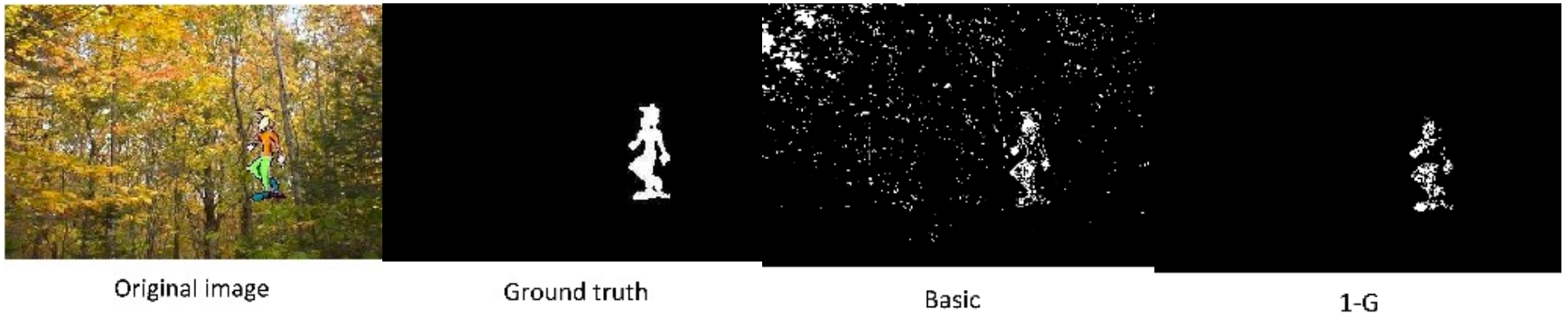
The Euclidean distance is thresholded to detect moving objects

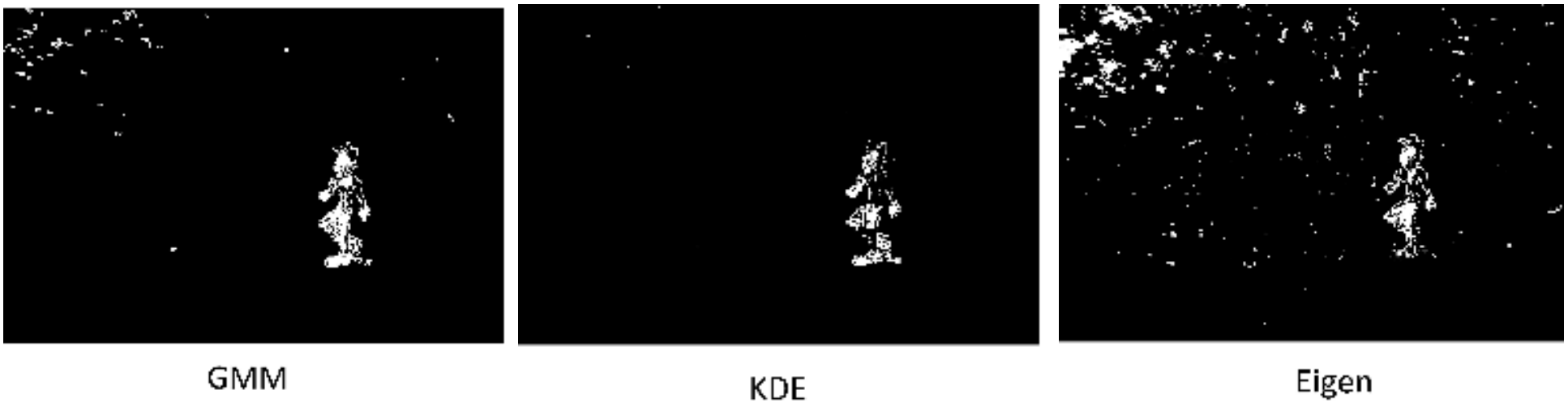
# Background subtraction. Eigenbackgrounds

1. The *n* frames are re-arranged as the columns of a matrix, A
2. The covariance matrix, $C = AA^T$, is computed
3. From *C*, the diagonal matrix of its eigenvalues, *L*, and the eigenvector matrix, $\Phi$, are computed
4. Only the first *M* eigenvectors (eigenbackgrounds) are retained
5. Once a new image, *I*, is available, it is first projected in the *M* eigenvectors sub-space and then reconstructed as *I'*
6. The difference *I* – *I'* is computed: since the sub-space well represents only the static parts of the scene, the outcome of this difference are the foreground objects

Slide credit: M. Piccardi

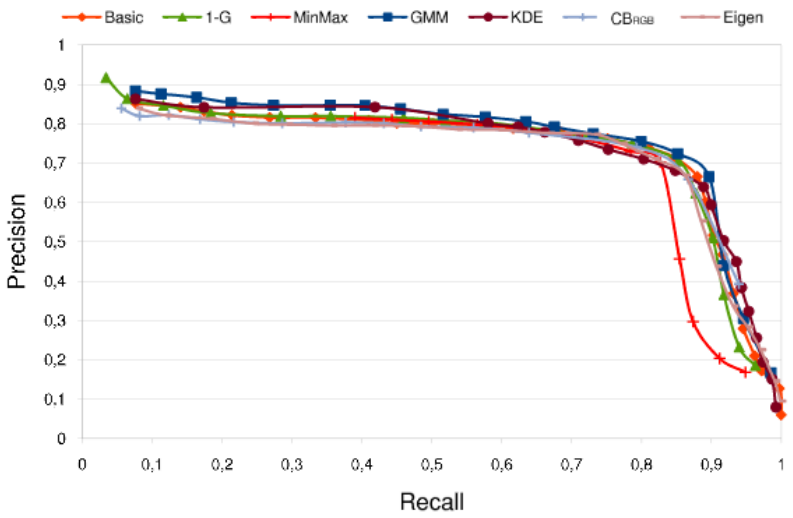
# Comparison



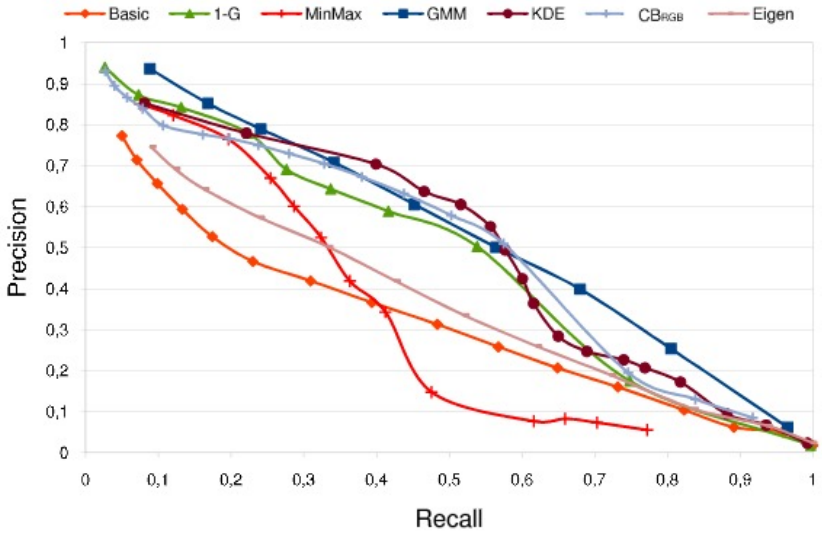




Benezeth et al., Comparative study… https://hal.inria.fr/inria-00545478/document

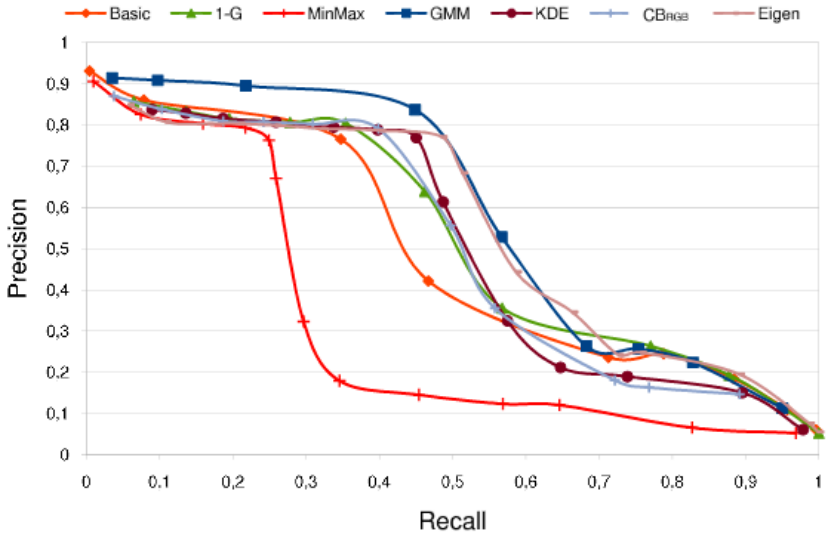
# Comparison



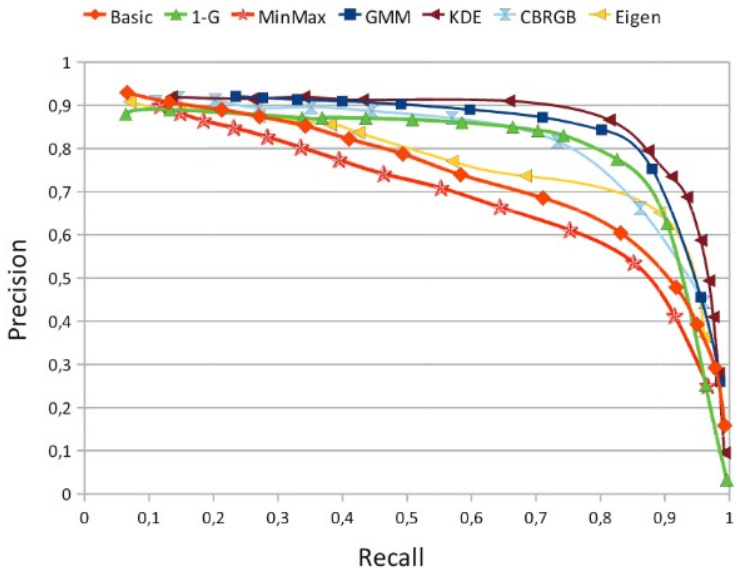

Noise-free videos with static backgrounds




Multimodal background



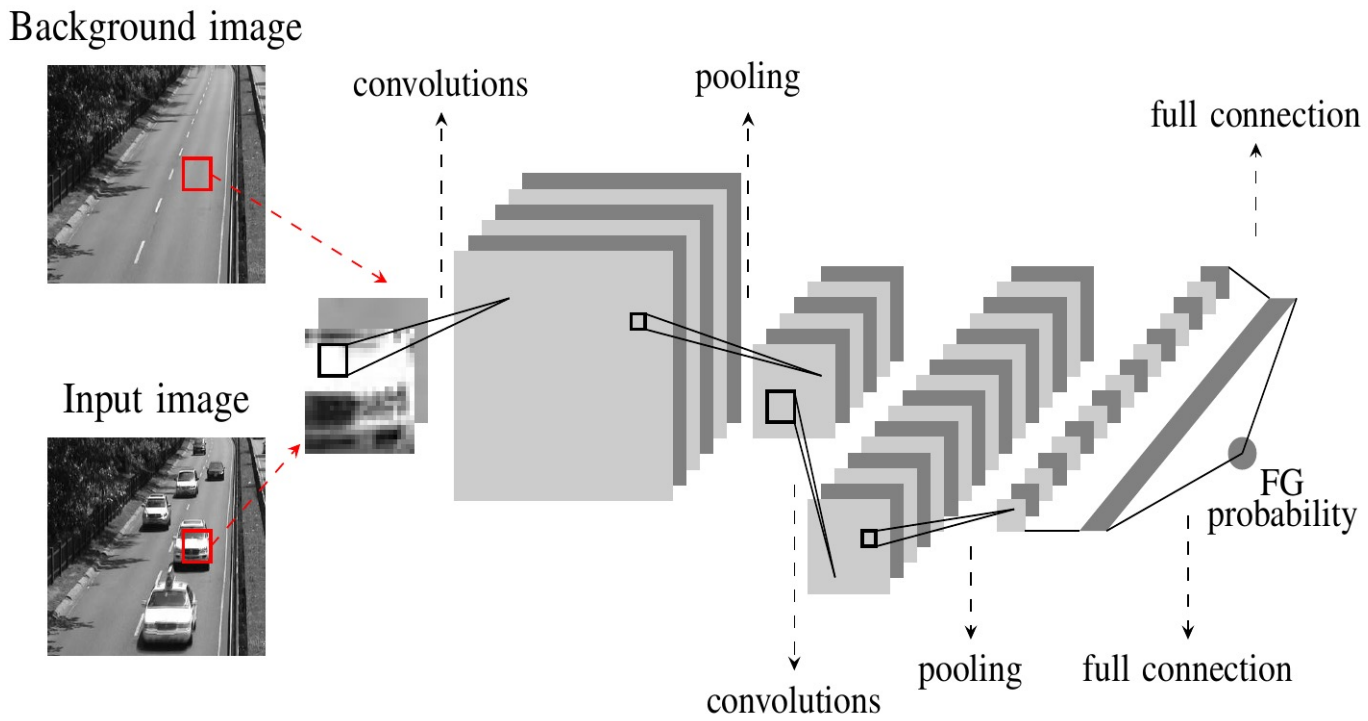

Noisy videos




Whole dataset with morphological filter post-proc

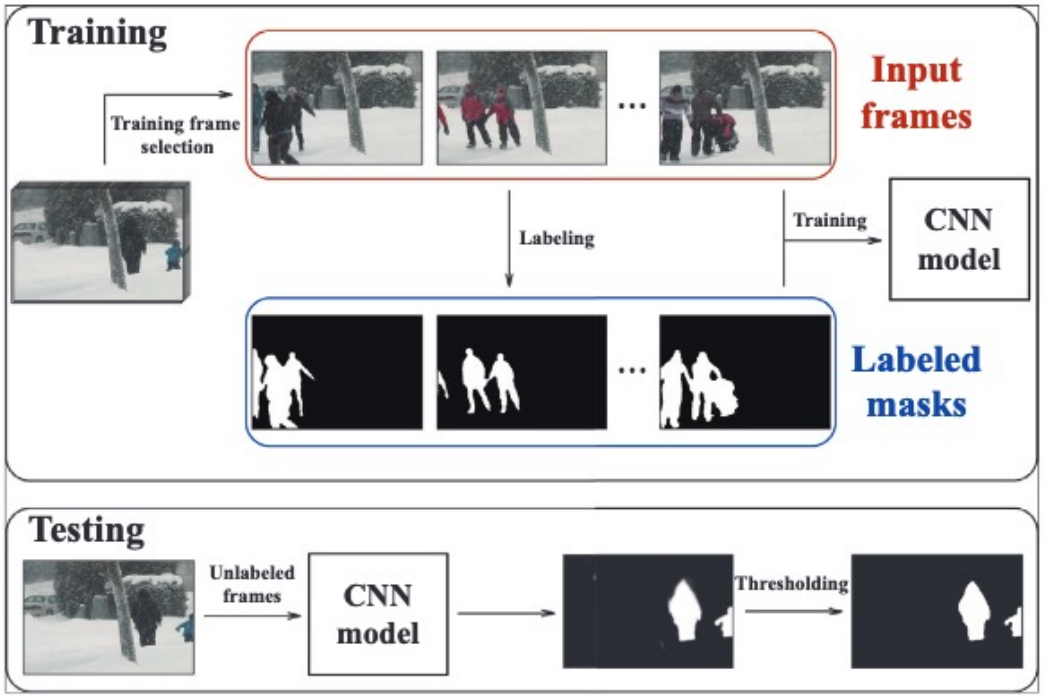
# Background subtraction with machine learning

- Applied for the classification step in “M. Braham et al. Deep Background Subtraction with Scene-Specific Convolutional Neural Networks. In IEEE IWSSIP, May 2016)”



# Background subtraction with machine learning

- CNNs trained on the current scene: “Wang, Y., Luo, Z., & Jodoin, P. M. (2017). Interactive deep learning method for segmenting moving objects. Pattern Recognition Letters, 96, 66-75.

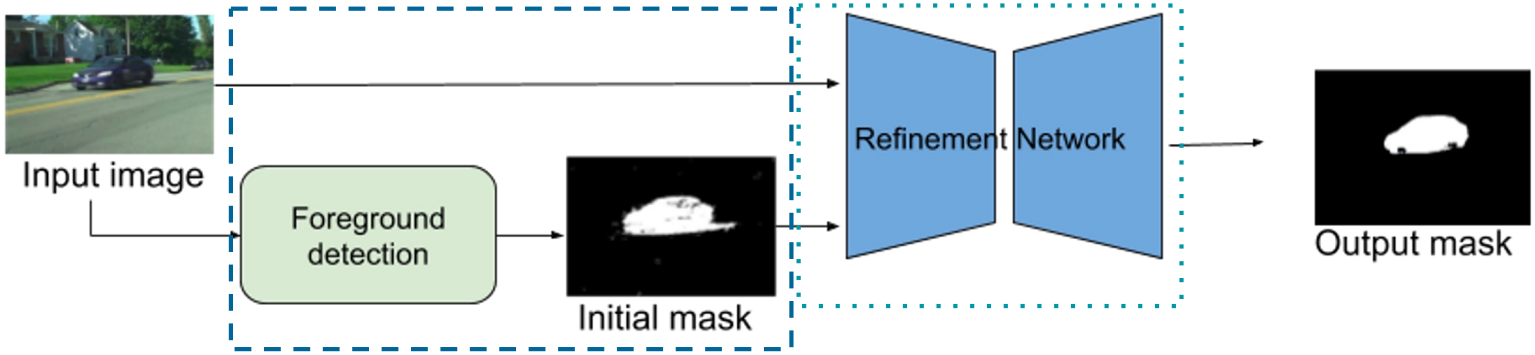


# Background subtraction with machine learning

- Refinement network, Pardàs, M., and Canet, G., "Refinement Network for unsupervised on the scene Foreground Segmentation." 2020 28th European Signal Processing Conference (EUSIPCO).

First Step:

- Conventional Background Subtraction method

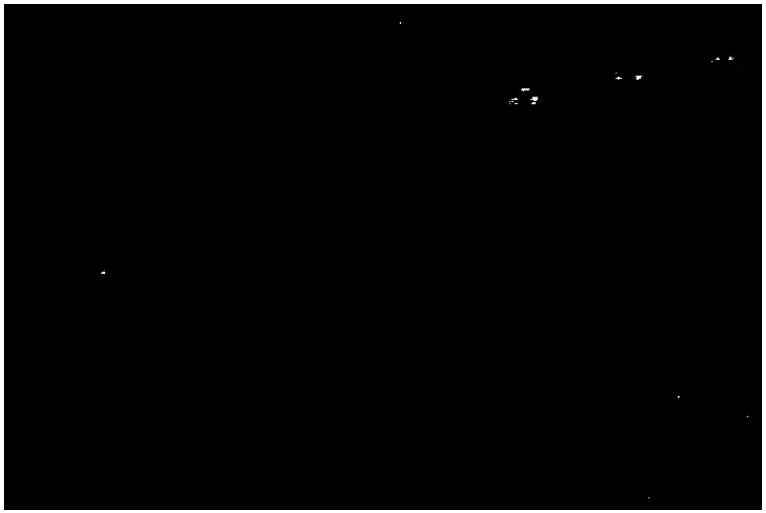



Second Step:

- Semantic Segmentation Network
- Trained to refine the previous foreground in a global scenario
- No specific training to the scene

# Examples

Original



Initial Mask: MOG2

Refinement Network

# Examples

Original

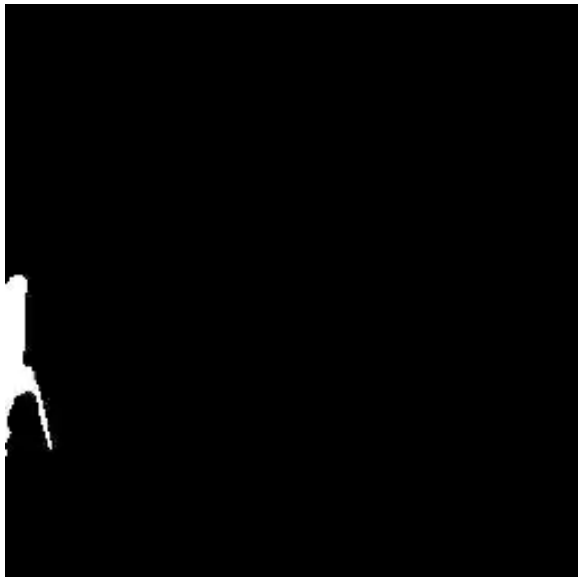
Initial Mask: MOG2

Refinement Network

# Examples

Original

Initial Mask: MOG2



Refinement Network

# Background subtraction with machine learning

- https://github.com/murari023/awesome-background-subtraction

# Shadow detection

- Cast shadows are an important problem in change detection
  - Shadow points are detectable as foreground points, since they differ from the background
  - Shadows have the same motion as the objects casting them

INCOMING IMAGE

ESTIMATED BACKGROUND

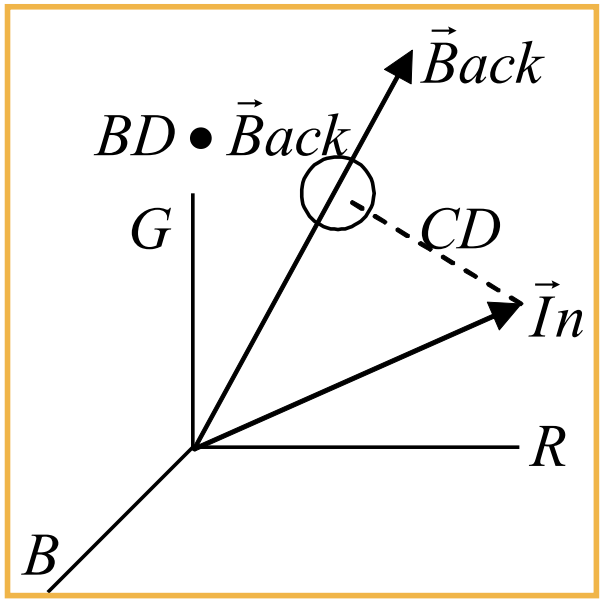
# Shadow detection

- Areas with shadows (highlightings) tend to have **similar chromaticity** as when not shadowed (highlighted) with lower (higher) brightness
- **Texture is also very similar** between shadowed and not shadowed regions.
- We can use our background representation to match those regions. A background image is commonly used.

# Shadow detection using BD and CD

Horprasert et al:

- **Brightness distortion** is a scalar that brings expected background colour close to the observed chromaticity line
- **Colour distortion** is the orthogonal distance between observed colour and expected chromaticity line




$$BD = \underset{\alpha}{\operatorname{argmin}} \left( \vec{I}n - \alpha \vec{B}ack \right)^2$$

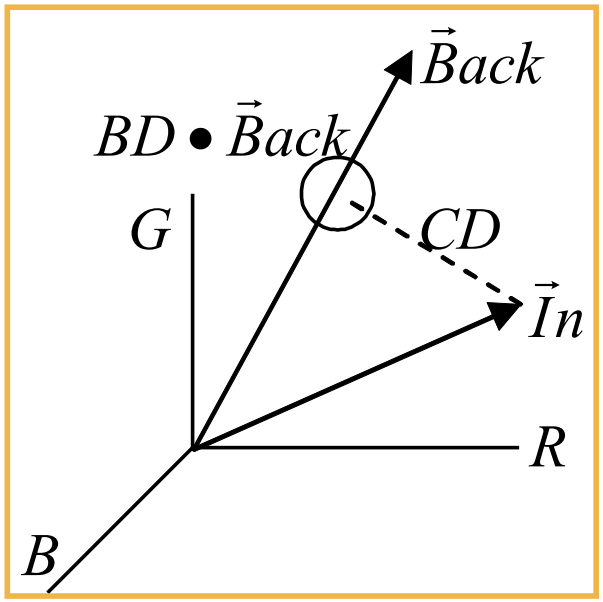
$$CD = \left\| \vec{I}n - BD.\vec{B}ack \right\|$$

# Shadow detection using BD and CD

Scalar projection = $S = \left|\vec{In}\right| \cos\theta = \vec{In} \bullet \frac{\vec{Back}}{\left|\vec{Back}\right|}$

Vector projection = $\vec{V} = S.\frac{\vec{Back}}{\left|\vec{Back}\right|} = BD.\vec{Back}$

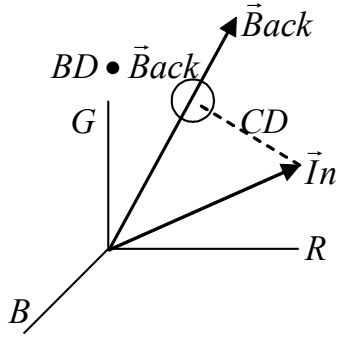
$$BD = \frac{S}{\left|\vec{Back}\right|} = \frac{\vec{In} \bullet \vec{Back}}{\left|\vec{Back}\right|^2}$$

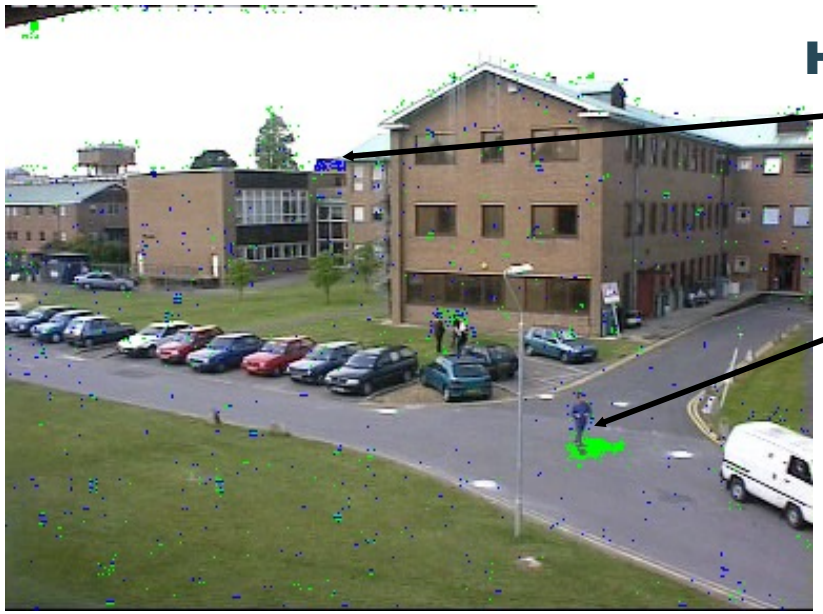


# Shadow detection using CD and BD: An example

```
IF CD < 10 THEN
    IF 1 > BD > 0.5 -> SHADOW
    IF 1.25 > BD > 1 -> HIGHLIGHTING

ELSE FOREGROUND
```

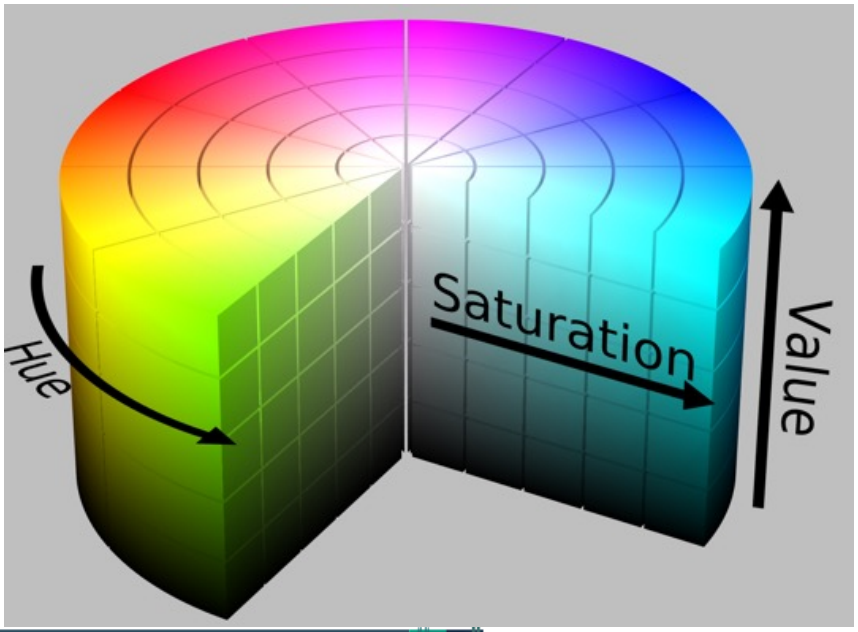


# Shadow detection using HSV

Cucchiara et al:
After Fg detection, generate Shadow mask SP

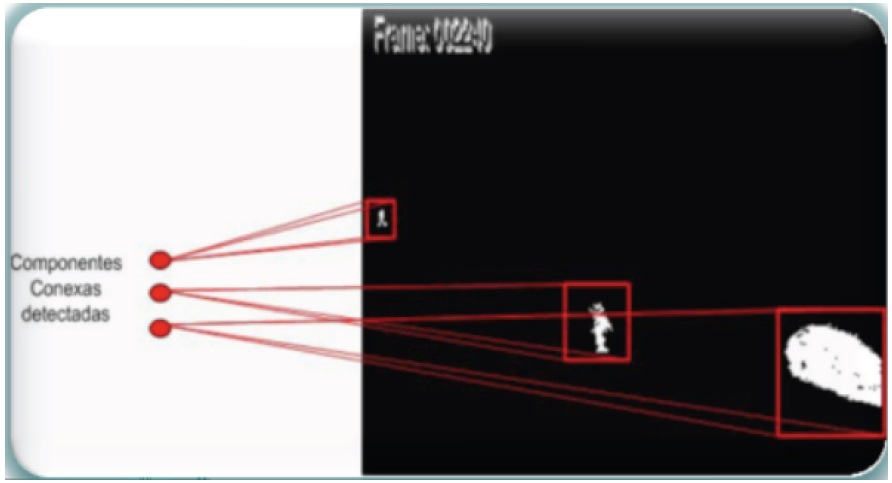
$$SP_k(x,y) = \begin{cases} 1 & if \quad \alpha \leq \frac{I_k^V(x,y)}{B_K^V(x,y)} \leq \beta \\ & \wedge (I_k^S(x,y) - B_k^S(x,y)) \leq \tau_S \\ & \wedge |I_k^H(x,y) - B_k^H(x,y)| \leq \tau_H \\ 0 & otherwise \end{cases}$$



# Tracking of segmented objects

Connected components based tracking

- In foreground segmentation methods, connected components detected must be associated to objects along time
- An object register is created that keeps information for each object: centroid, histrogram, velocity, size,..
  - Connected components detected and tracked for $T > T_{min}$ are associated to a new object
  - Objects not tracked for $T > T_{lost}$ are removed

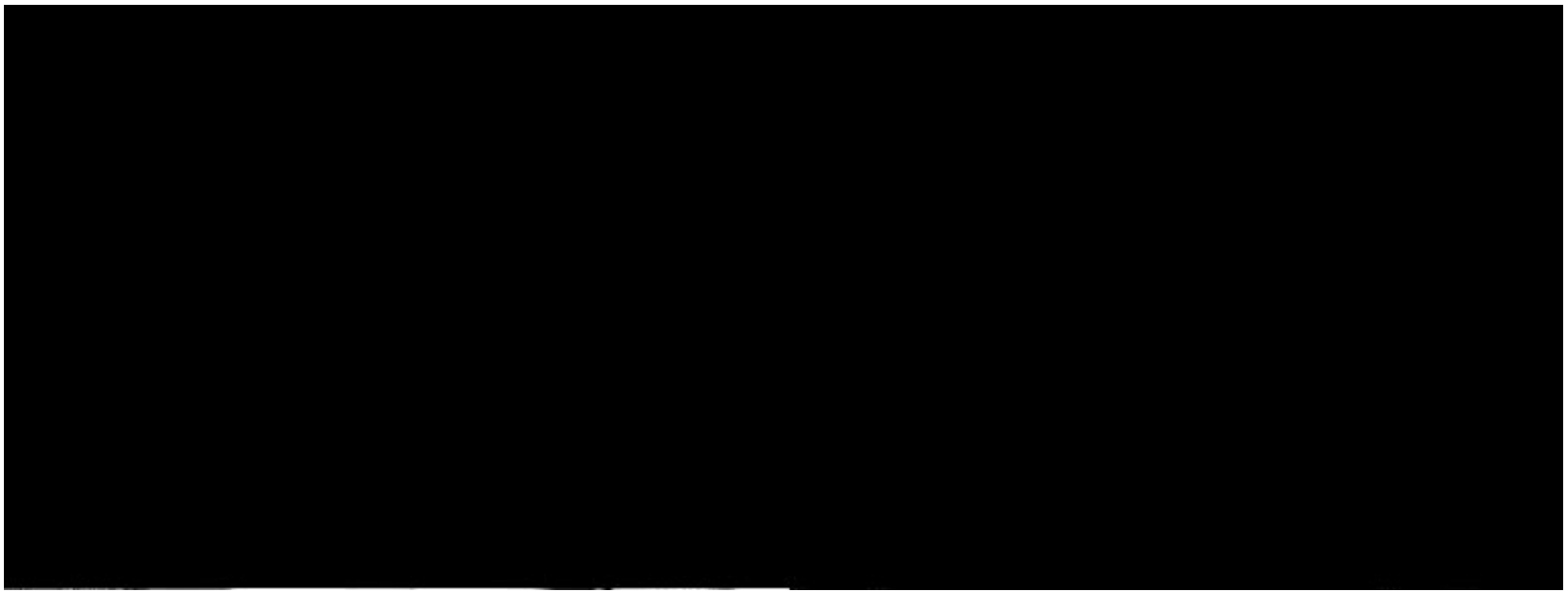


# Examples

# Tracking of segmented objects

- Apply foreground detection
- Identify Connected Components
- Apply tracking method to each object, using only foreground pixels
- Associate multiple CC to an object: all the CC within a rectangular area of the size of the object around the predicted centroid position
- Update object features:
  - No collision: The object does not share CC with other objects. Update object's size, position and histogram
  - Collision: The object does share CC with other objects. Only update object's position with the estimation of the tracking
- Initialize and remove objects

# Examples



# Further problems in security applications

- When large areas have to be monitored, **cameras may not be static** but may scan a given area:

✓ Images are compared against a **panoramic view of the scene**. The system has to know where the camera is pointing to at every instant.

# Bibliography


- M. Piccardi. Background subtraction techniques: a review. In IEEE Int. Conf. On Systems, Man and Cybernetics 2004 , v. 4, pp. 3099-3104, 2004.
- Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In IEEE CVPR'99, volume 1, pages 22–29, 1999.
- A. Elgammal, D. Harwood, and L. S. Davis. Nonparametric background model for background subtraction. In ECCV'00, pages 751–767, Dublin, 2000.
- Oliver, Rosario, Pentland, A Bayesian Computer Vision System for modeling Human Interactions, IEEE PAMI, 2000.
- T. Horprasert, D. Harwood, and L.S. Davis, "A Statistical Approach for Real-Time Robust Background Subtraction and Shadow Detection," Proc. IEEE Int'l Conf. Computer Vision '99 FRAME-RATE Workshop, 1999.
- Andrea Prati, IvanaMikic, Mohan M. Trivedi, and Rita Cucchiara. Detecting moving shadows: Algorithms and evaluation. IEEE Trans. Pattern Anal. Mach. Intell., 25(7):918– 923, 2003.
- R. Cucchiara, C. Grana, M. Piccardi, A. Prati, and S. Sirotti. Improving shadow suppression in moving object detection with hsv color information. In Proceedings. 2001 IEEE Intelligent Transportation Systems, pages 334–339, 2001.
- **Scene background modeling contest** http://pione.dinf.usherbrooke.ca/sbmc2016/
- **Changedetection.net**

# Outline

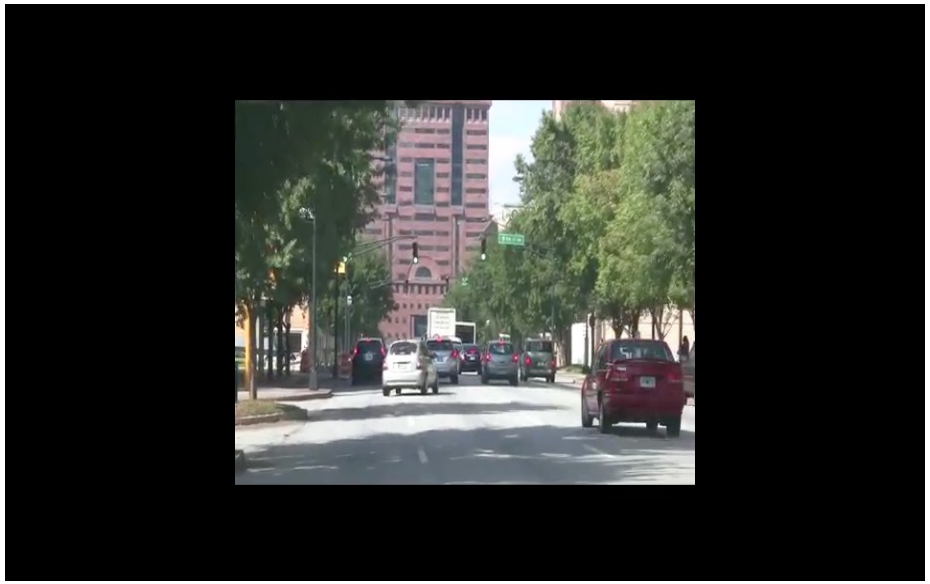
- Shot detection
- Moving object segmentation
- Region segmentation
  - Spatial segmentation and tracking
    - Possible features: Motion, texture, depth
  - Spatio-temporal segmentation

# 3D region segmentation

www.videosegmentation.com

# 3D region segmentation

- Applications of video understanding:
  - Video indexing and search: regions must retain enough semantic information for classification tasks
    - Selecting regions⇒ rapid annotation
    - Spatio-temporal region classification
    - Content-based retrieval
  - Activity recognition
  - Sports analysis
  - Advertising analytics
  - Robotics



# Introduction to video segmentation

- The division in three steps proposed for image segmentation is adopted:


- **Remove useless or annoying information**
- **Preserve shape information**

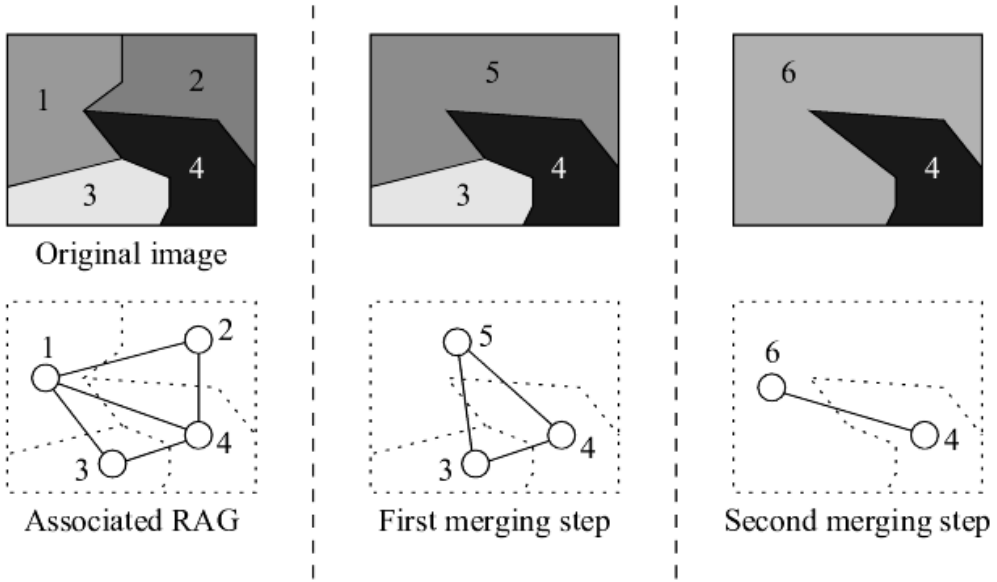
- **Feature space**:
  Gray level/Color
  Texture
  Motion
  Depth
  Frame difference
  DFD
  Histogram
- Elements belonging to the same regions should correspond to **homogeneous features**

- **Decision space**:
  1D: Temporal
  2D: Spatial
  3D: Spatial/temporal
- **Partition definition**:
  - ✓ Transition
  - ✓ Homogeneity

# Hierarchical motion/color segmentation

- An **object** is a set of **color homogenous regions that jointly move, following a given motion model.**

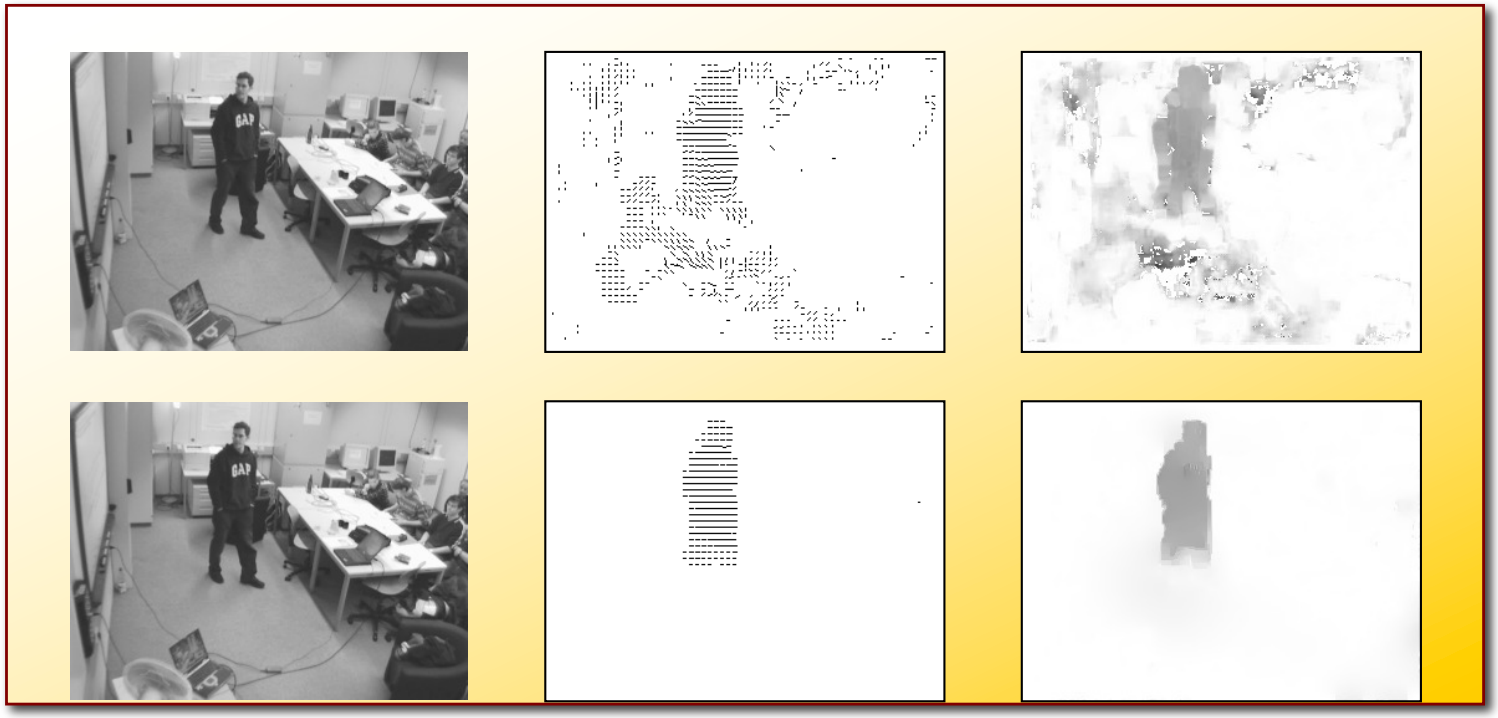
- Iterative merging process:
  - Define an initial partition and its RAG (nodes: regions, edges: neighboring region similarity)
  - Find the pair of most similar neighboring regions
  - Merge them
  - Update the edges



# Hierarchical motion/color segmentation

Feature: Motion

- Estimate a dense motion field (Optical flow) and segment it.
  - Motion information has to be estimated:
    Contours are not well defined.
  - Optical flow may be very noisy:
    Robust estimators

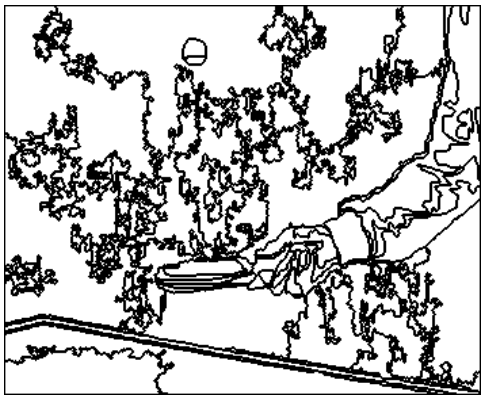


# Hierarchical motion/color segmentation

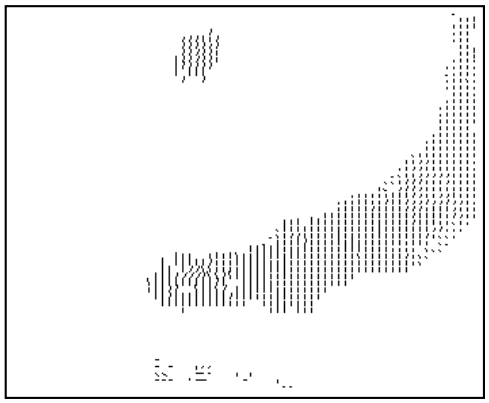
Features: texture + motion

- Define an initial color-based segmentation (using color/contour similarity measures)
- Estimate the motion information between neighboring frames

**Image #n**



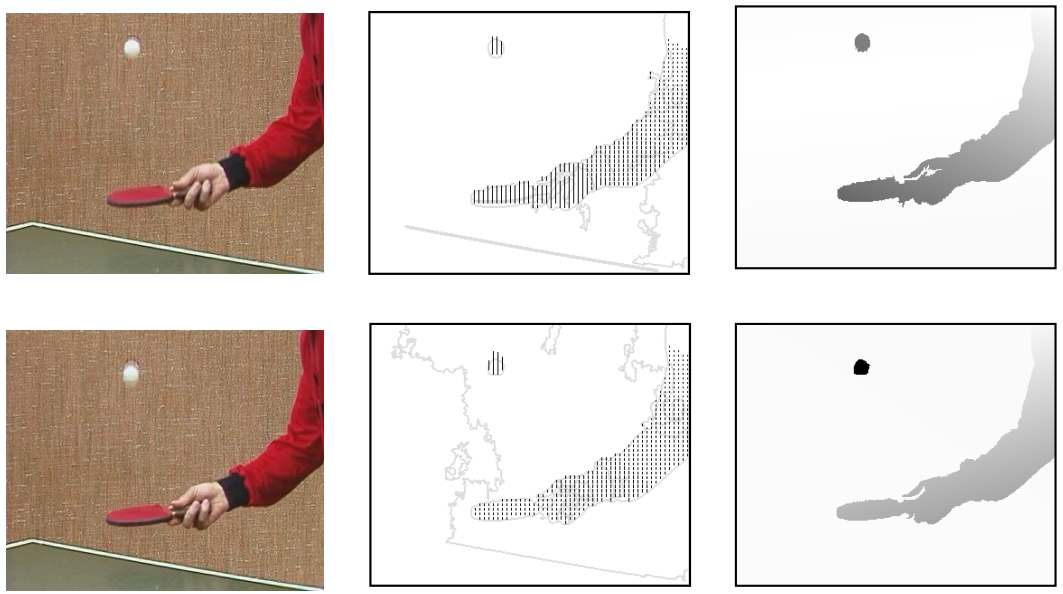
**Texture Partition of image #n**



**Vector field between images #n and #(n+1)**

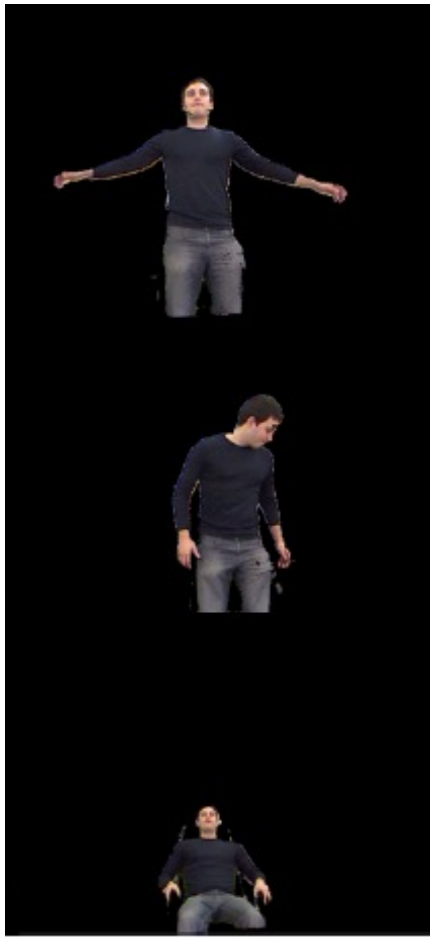
# Hierarchical motion/color segmentation

- Once regions cannot be further merged (because they are not homogeneous anymore in terms of color), merging is performed based on motion information



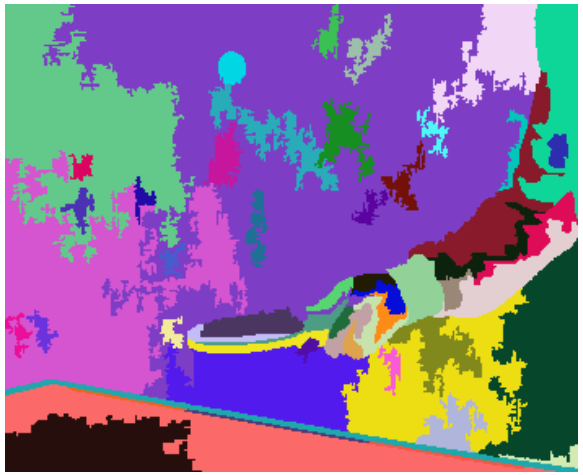
Color partition | Motion estimation | Motion partition

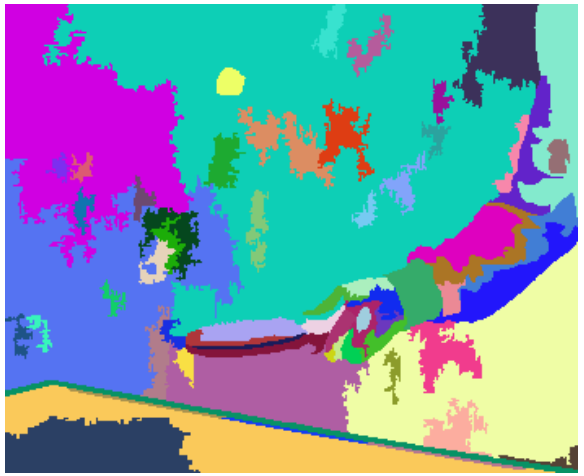
# Hierarchical depth/color segmentation



# Spatial segmentation and tracking

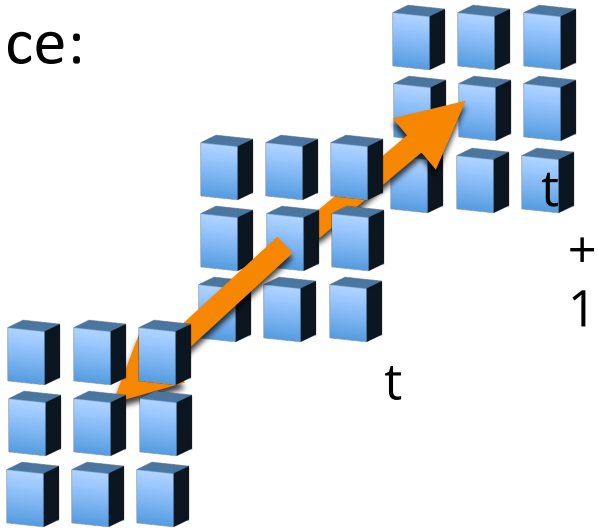
– To obtain the temporal link between regions a region matching step is necessary:
  - Region descriptors such as their position (original and after motion compensation), color, texture, size, shape, …
  - Structure descriptors such as the relative position among different regions: subgraph matching techniques



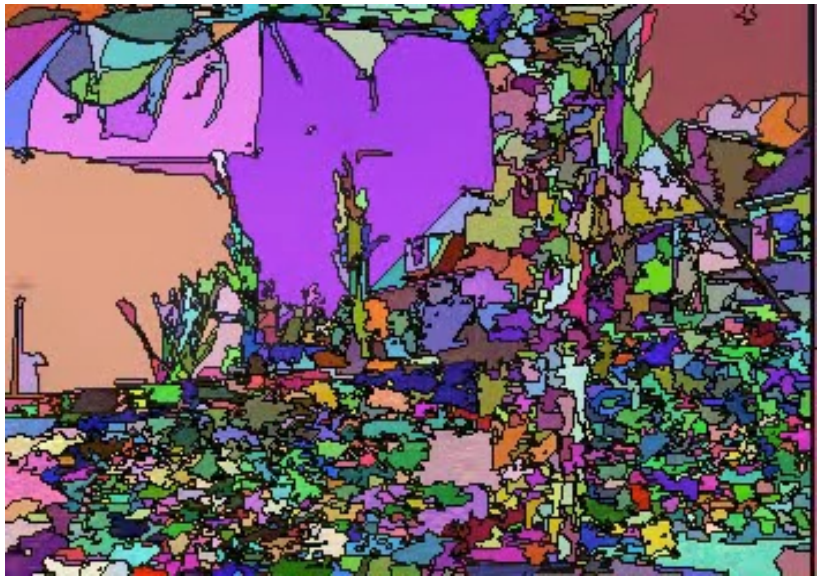


# Spatio-temporal segmentation (3D regions)

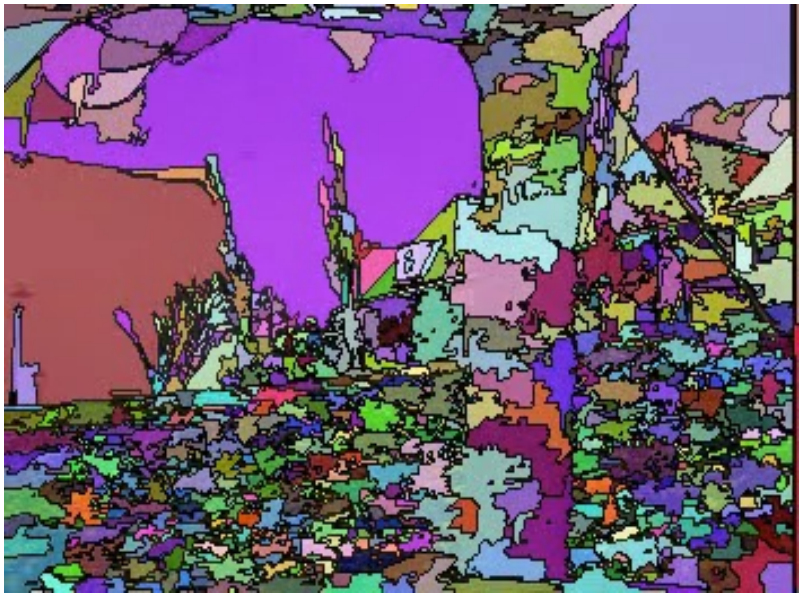
- Segmentation algorithms require the definition of neighborhoods.
- Neighborhoods can be defined in the 3D space:
  - Spatial connectivity:
    - Four- or Eight-connected neighborhoods
  - Temporal connectivity:
    - Collocated pixels in subsequent images:
      Object velocity versus object size:
      Instances of the same object disconnected through time
      It may require an object/region matching algorithm
    - Pixels related through motion estimation:
      Motion has to be estimated:
      Inaccurate connectivity between spatial components
      It may require defining uncertainty areas



# Spatio-temporal segmentation



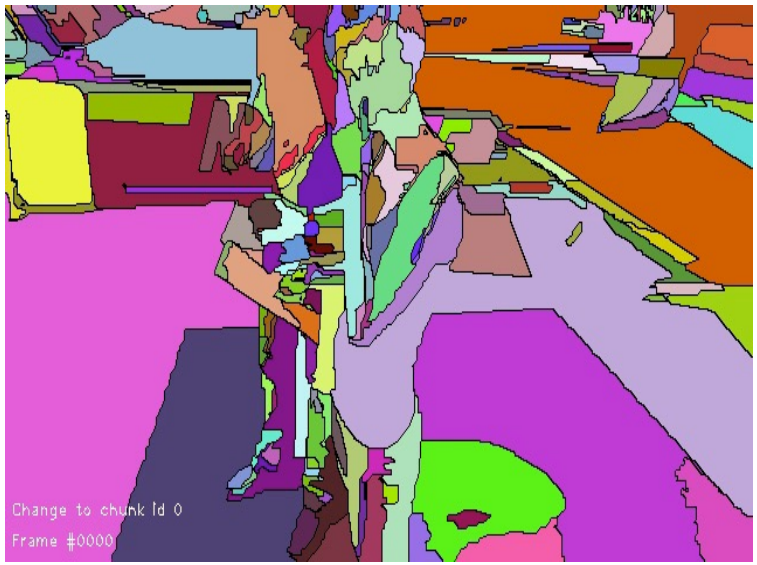
oversegmentation using predecessor along dense flow



oversegmentation using direct predecessor in volume

From: Graph-Based Hierarchical Video Segmentation
M. Grundmann, V. Kwatra, M. Han (Google Research)
D. Castro, I. Essa (Georgia Institute of Technology)

# Spatio-temporal segmentation

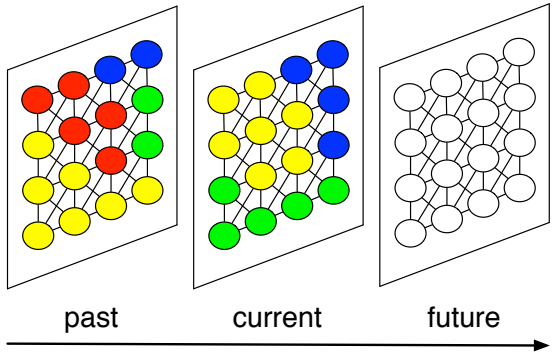



oversegmentation using predecessor along dense flow


oversegmentation using direct predecessor in volume

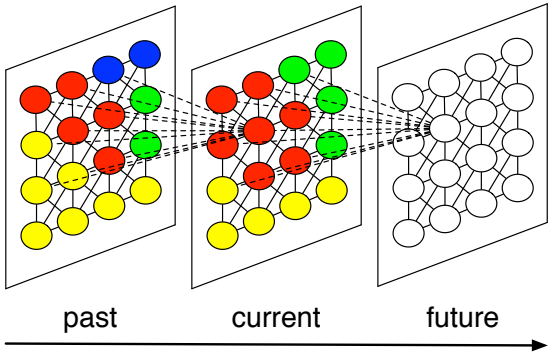
From: Graph-Based Hierarchical Video Segmentation
M. Grundmann, V. Kwatra, M. Han (Google Research)
D. Castro, I. Essa (Georgia Institute of Technology)

# Spatio-temporal segmentation



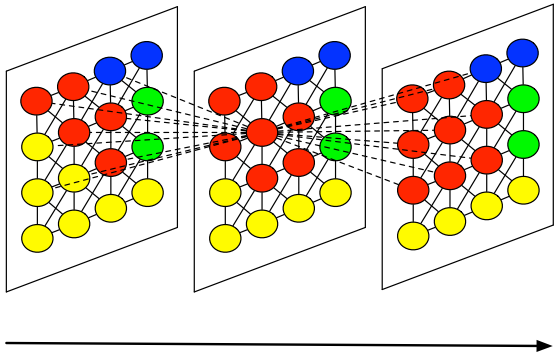

Frame-by-Frame

[Brendel and Todorovic ICCV 2009]
[Lee et al. CVPR 2011]




Streaming - Time recursive

[Paris ECCV 2008]
[Grundmann et al. CVPR 2010] (Clip-based)



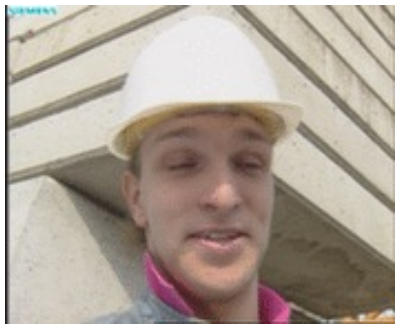
Full Video

[Paris and Durand CVPR 2007]
[Grundmann et al. CVPR 2010]
[Lezama et al. CVPR 2011]

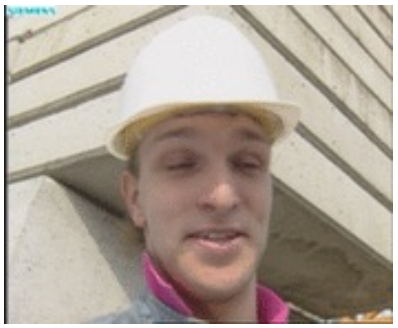
Slide credit: J. Corso

# 2D+Time region growing: a possible implementation (I)

- Define a segmentation of the image at time t.
- Estimate the motion information between neighboring frames.
- Make a prediction of the partition at time t+1 using the partition at time t and the motion information (tracking is done).
- Check if new regions have to be created.
- This will create many uncertainty areas: may be solved by **region growing**.

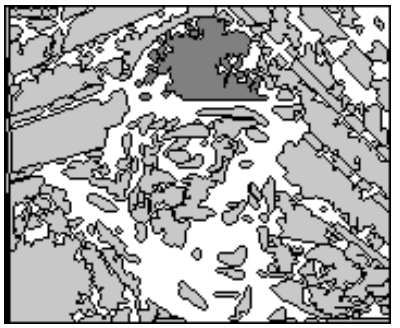


**Frame #0**



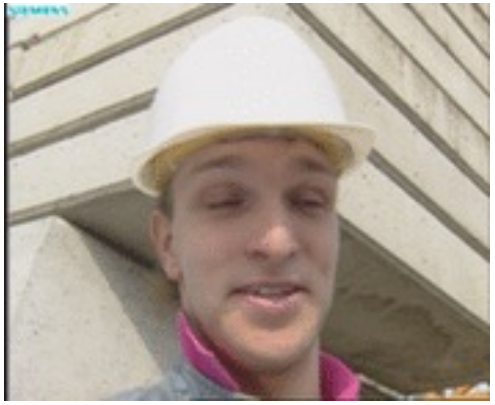
**Frame #1**

**Partition at #0**

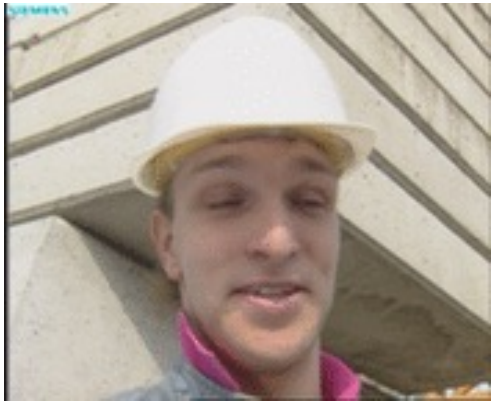


**Predicted partition**

# 2D+Time region growing: a possible implementation (II)

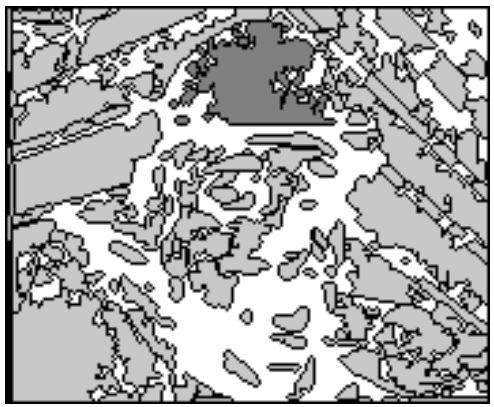


Frame #0



Frame #1

Partition at #0

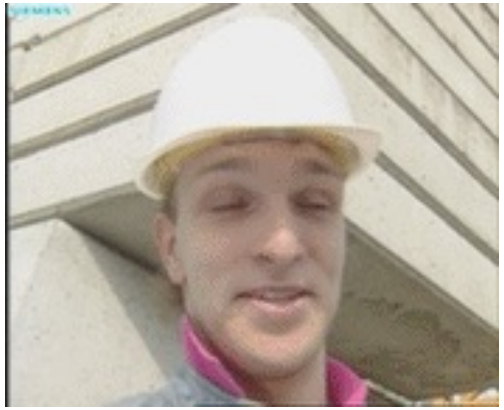


Predicted partition

= Markers

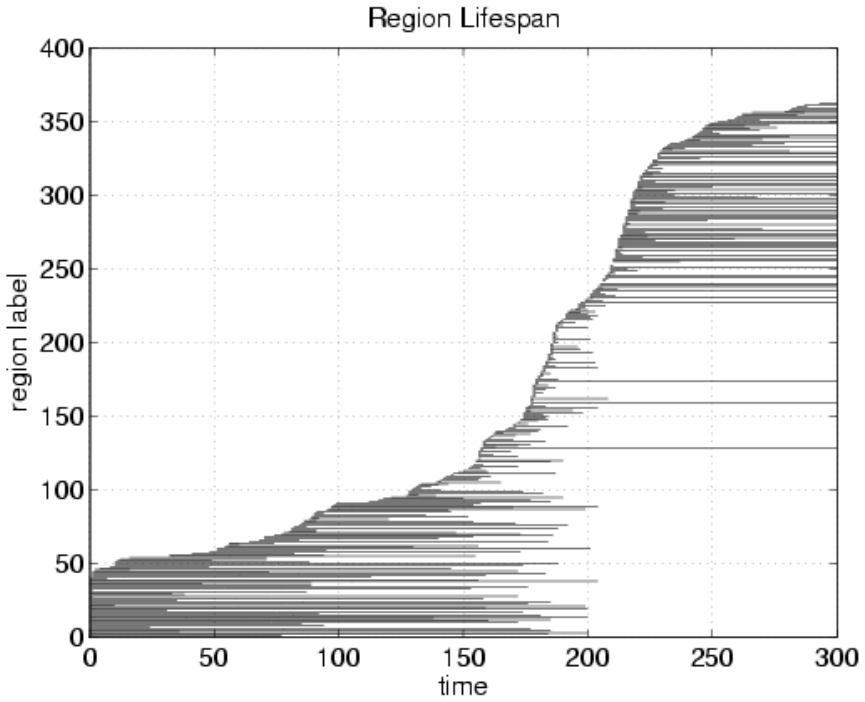
Partition at #1

# 2D+Time region growing: a possible implementation (III)



**Original sequence**

**Partition sequence**



# Bibliography


- M. Grundmann, V. Kwatra, M. Han, I. Essa, Efficient hierarchical graph-based segmentation
- C. Xu, C. Xiong, and J. J. Corso. Streaming hierarchical video segmentation. In *Proc. of European Conf. on Computer Vision*, 2012
- Galasso et al. , Video Segmentation with superpixels, ACCV 12
- Chang et al. , Temporal Superpixels, CVPR 13
- Van der Bergh et al., Video Seeds, ICCV 13
- Palou and Salembier, Trajectory Binary Partition Tree, CVPR13
- www.supervoxel.com