# Motion segmentation—separating background and foreground

**Tomáš Svoboda**, svoboda@cmp.felk.cvut.cz
Czech Technical University in Prague, Center for Machine Perception
http://cmp.felk.cvut.cz
**Last update:** December 3, 2007

## Talk Outline

- Importance
- Challenges
- Solutions

## Scenario

- static cameras observe a room, street, parking, . . .
- (almost) static background—the scene
- moving object (cars) and/or people—foreground

## The Goal

- detect meaningful motions
- avoid false detection caused by unimportant changes in the scene (image noise, swaying trees, illumination changes, shadows, . . . )

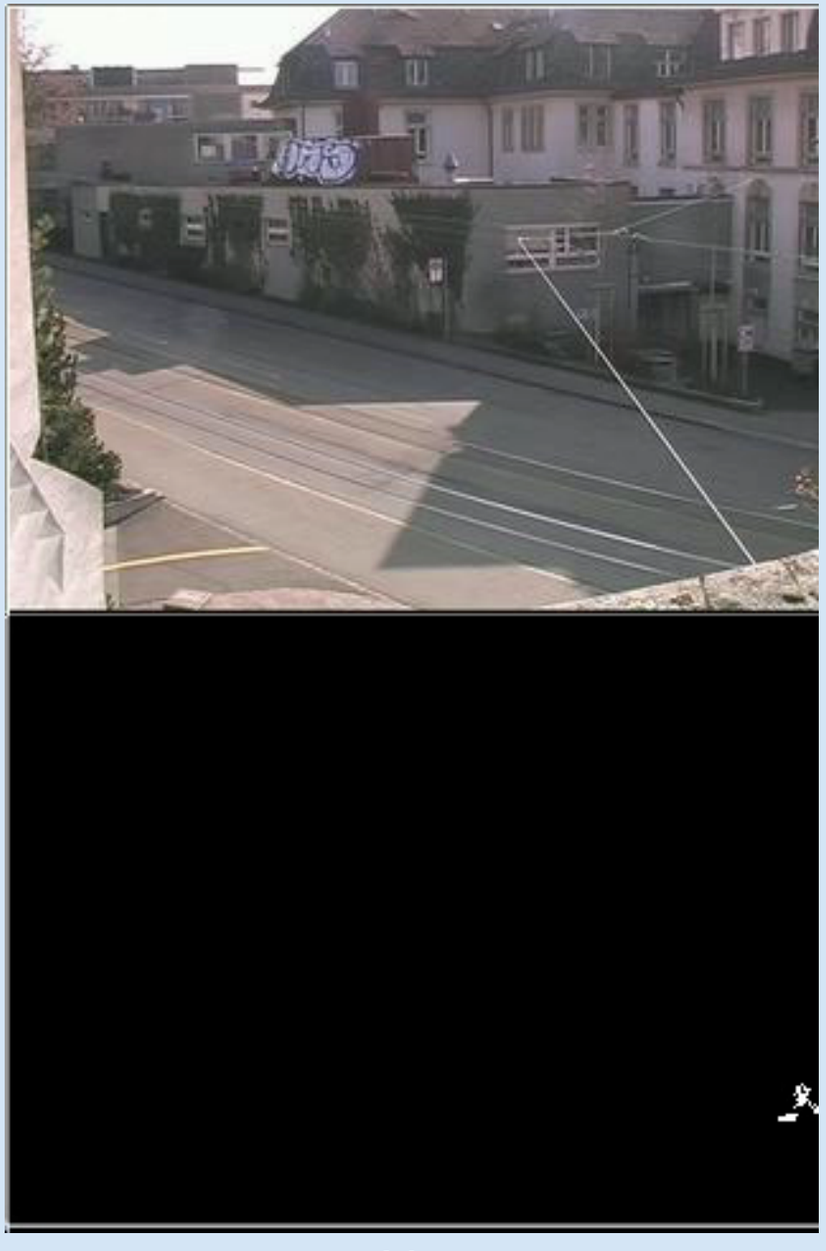
# Applications

- surveillance (detect: intruder in forbidden zone, panic situation, unusual behavior, long queues . . . ; alert human operator)
- human-computer interaction without input devices
- tele-presence (tele-teaching, tele-conferencing . . . ) in larger scenes
- collision avoidance for robot/vehicle navigation
- video indexing: e.g. show me scenes in the movie with actor X
- . . .

Focus is on humans, but it is applicable to other objects as well.

# Application example — surveillance

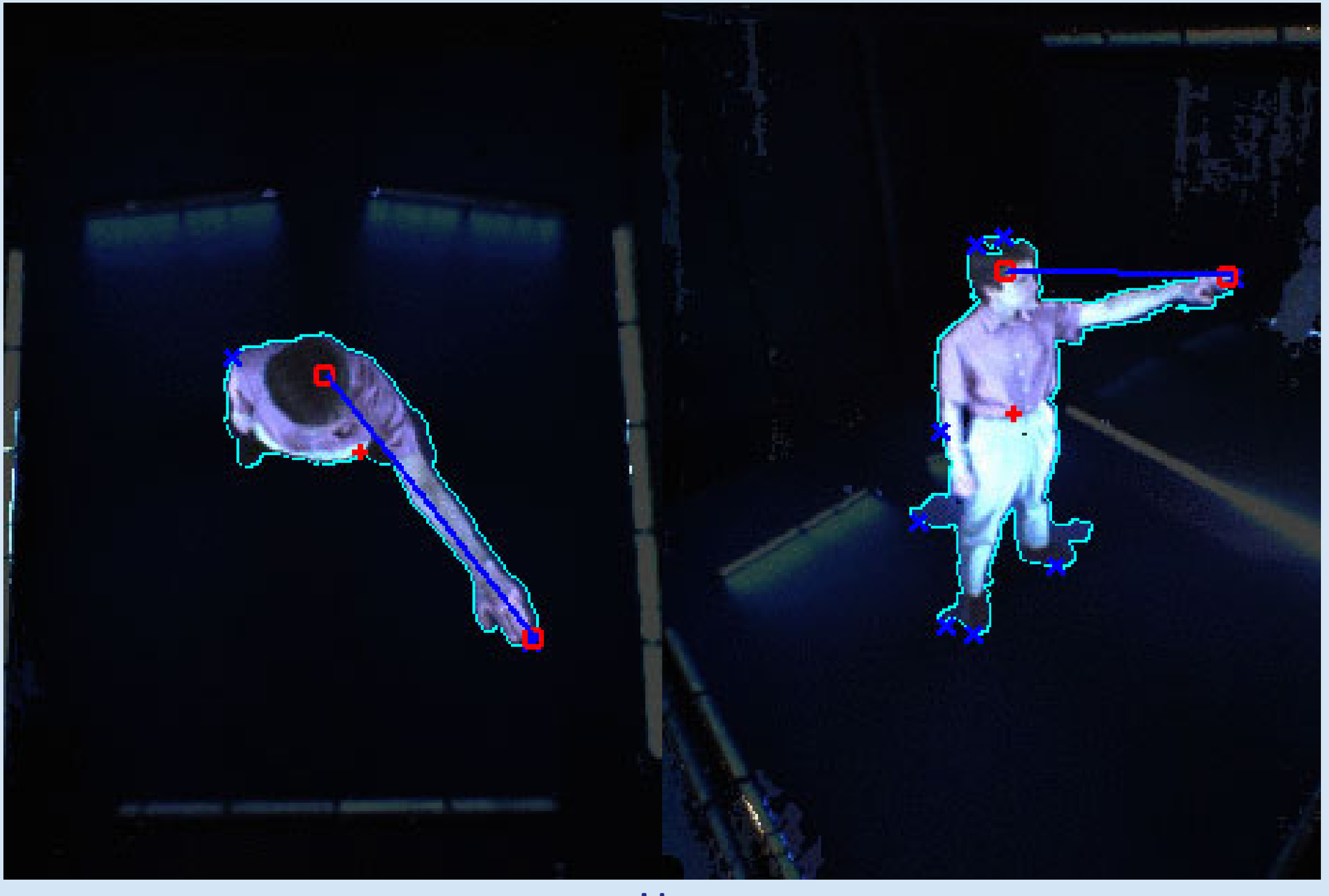




video

# Application example — gesture recognition for virtual environments



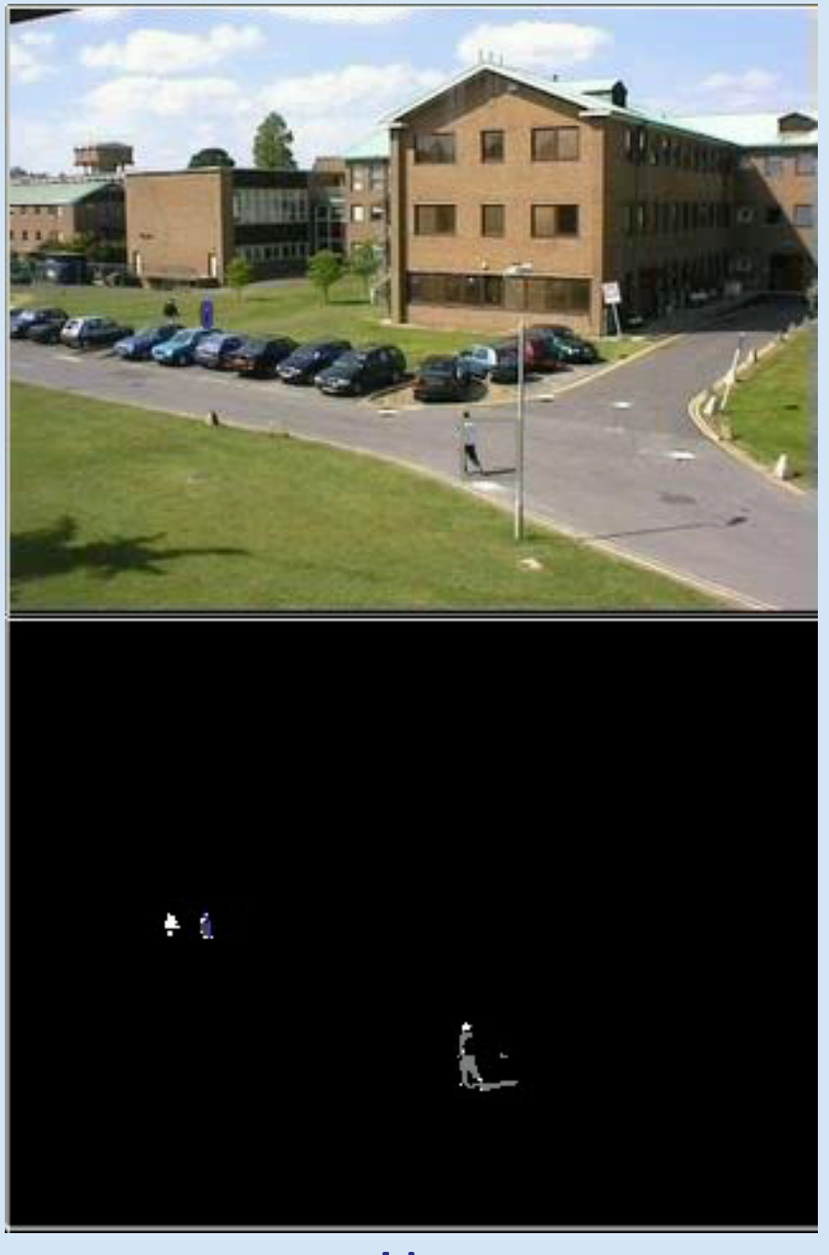


video

# Challenges

- non-interesting changes in image
- shadows
- appearance changes (rotating car)
- objects that temporarily stopped
- overlapping objects
- moving camera (moving background)
- articulated objects

For us (humans), this is easy, but even the best computer vision methods work only under some assumptions. But computers are cheaper, usually faster and stay concentrated even when task is boring.



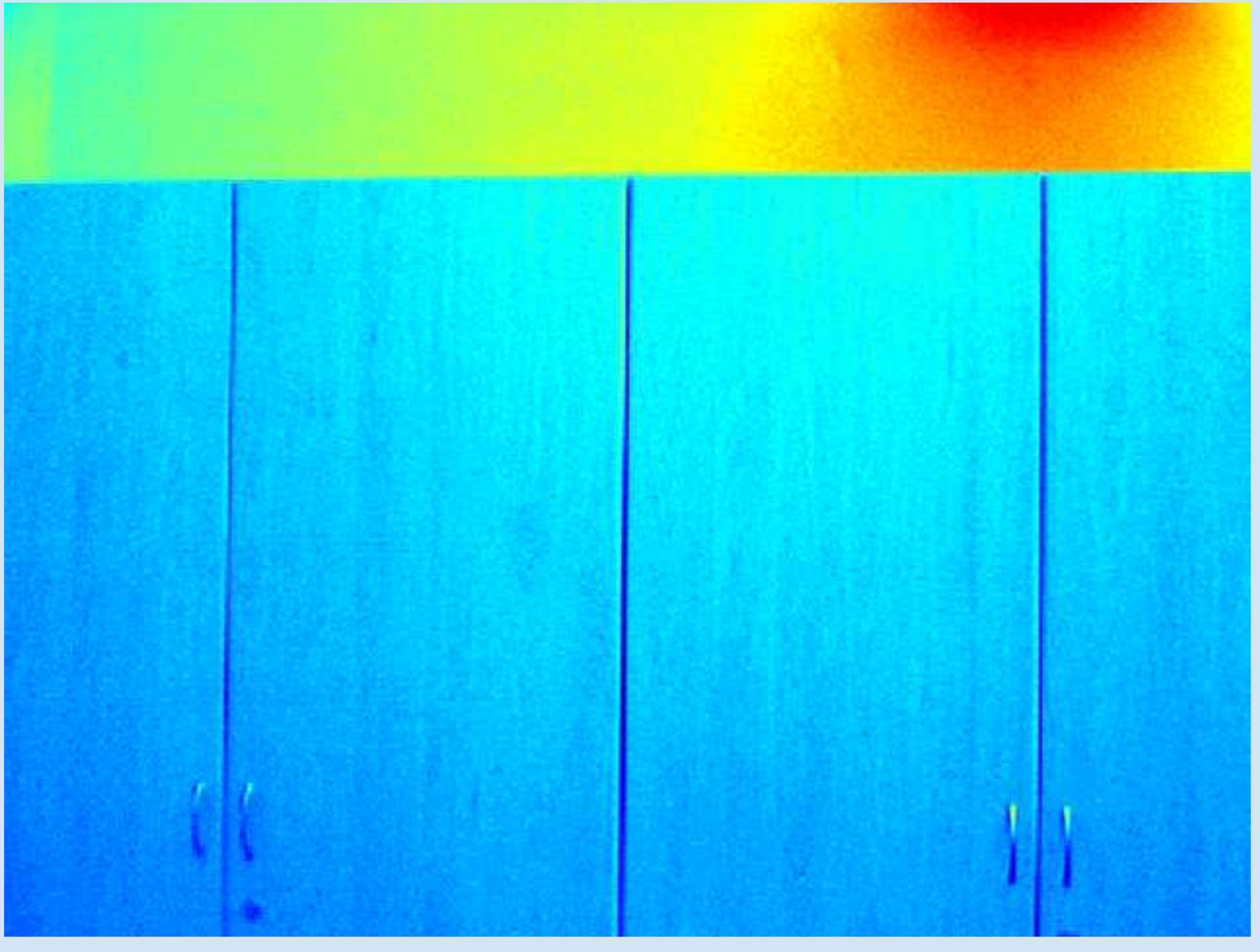
video

# Image noise

video

# Image noise (pseudocolors of red channel)

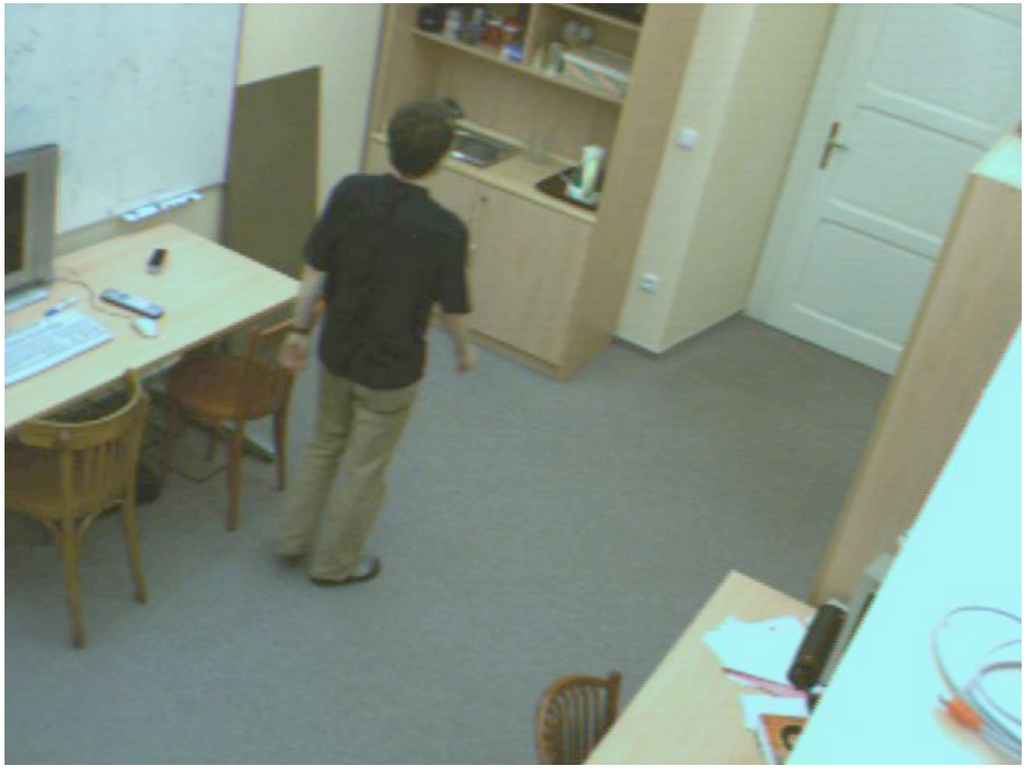





video

# Essentials

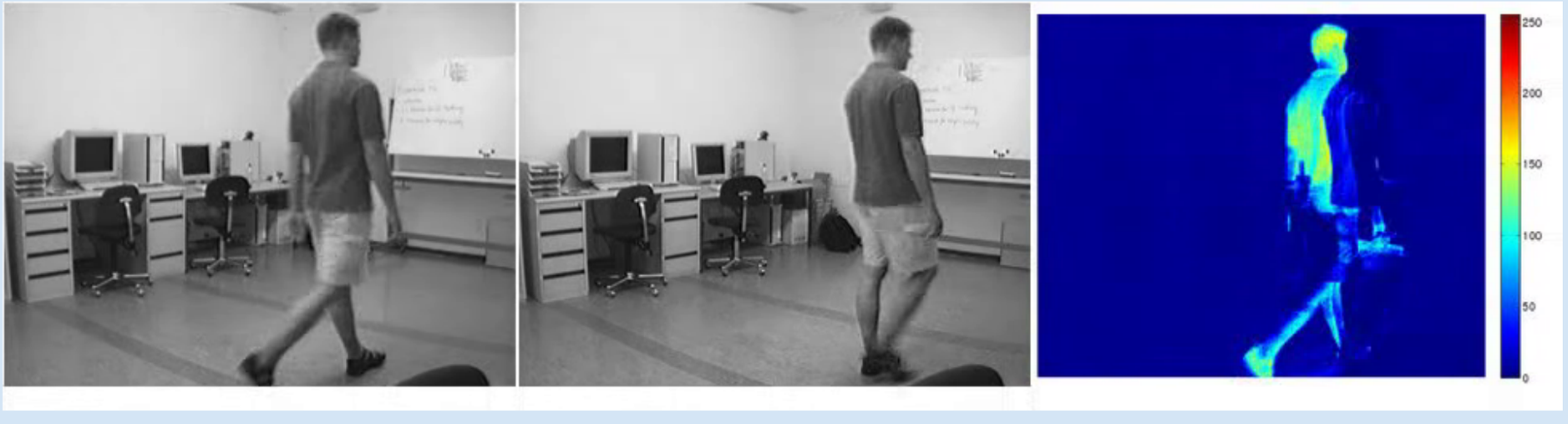
- no high-level knowledge about the scene
- pixel based classification
- observations and some rough assumptions about the scene only available



# Possible solution I: Frame differencing

**Idea:** compare subsequent frames

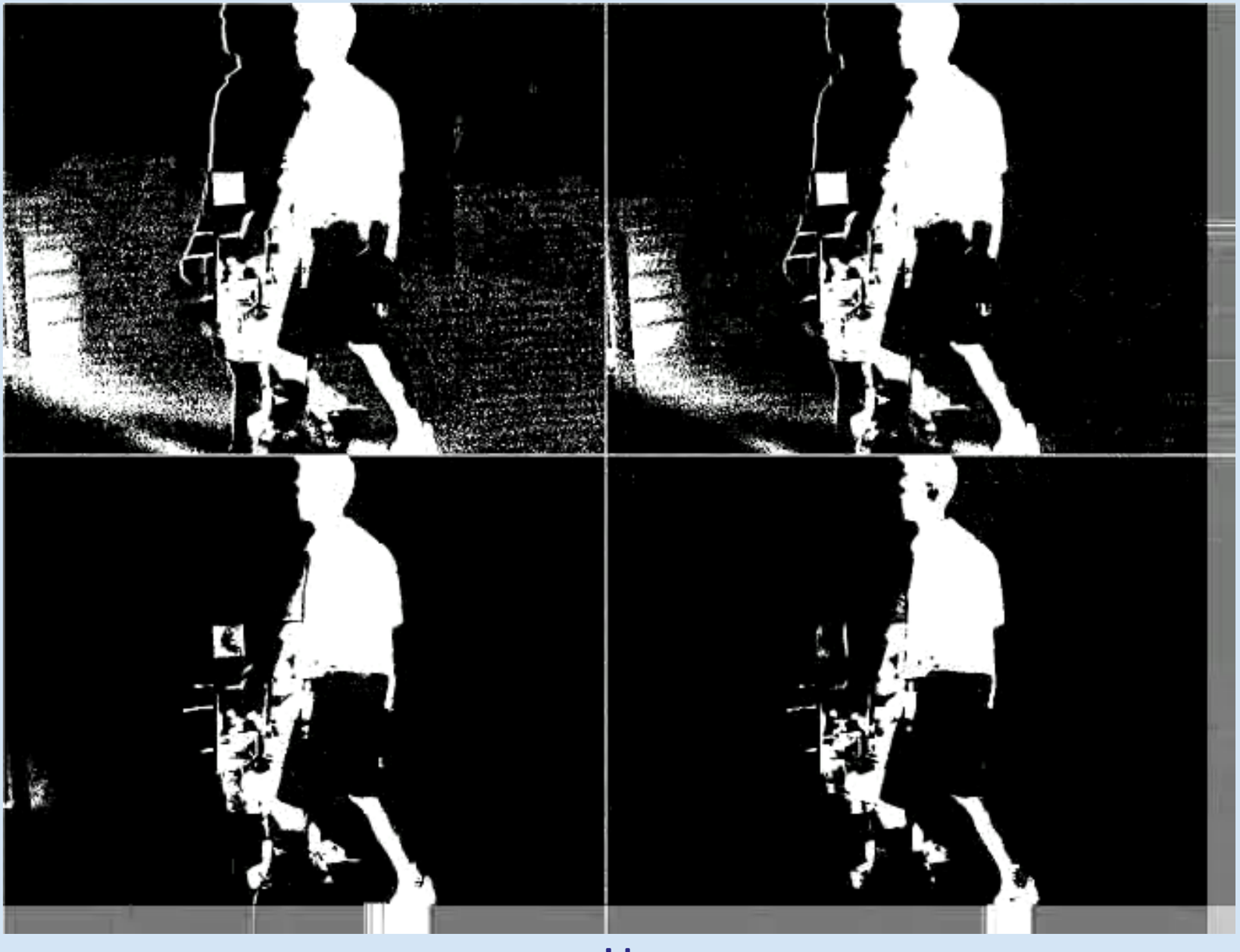
Pixel is part of foreground (object) if $\|I_t - I_{t-1}\| > T$ where $T$ is a threshold




**video**

# Frame differencing — thresholding





video

# Possible solution II: Background subtraction

**Idea:** learn the static part of the scene and always compare to actual frames

Pixel is part of foreground (object) if $\|I_t - B\| > T$ where $T$ is a threshold and $B$ is the background intensity.

**How to find $B$**

- just take a sample image
- capture sample images and use the average

**How to find $T$**

- test and try
- statistical analysis of a training set

# Naive solution

For each pixel, store the intensity $I_1$ observed in first image frame as background model $B = I_1$ and label the same pixel at frame $t$ as background if

$$|I_t - B| < T \ .$$

Problem: even image of completely static scene is not constant due to noise.

- taking background model from single image is not reliable – use mean image of $n$ first frames of empty scene $B = \mathrm{mean}\{I_t : t = 1 \ldots n\}$
- noise level can differ based on camera, its settings and illumination – use significance test to set the $T$ (Global or per-pixel)

# Background image

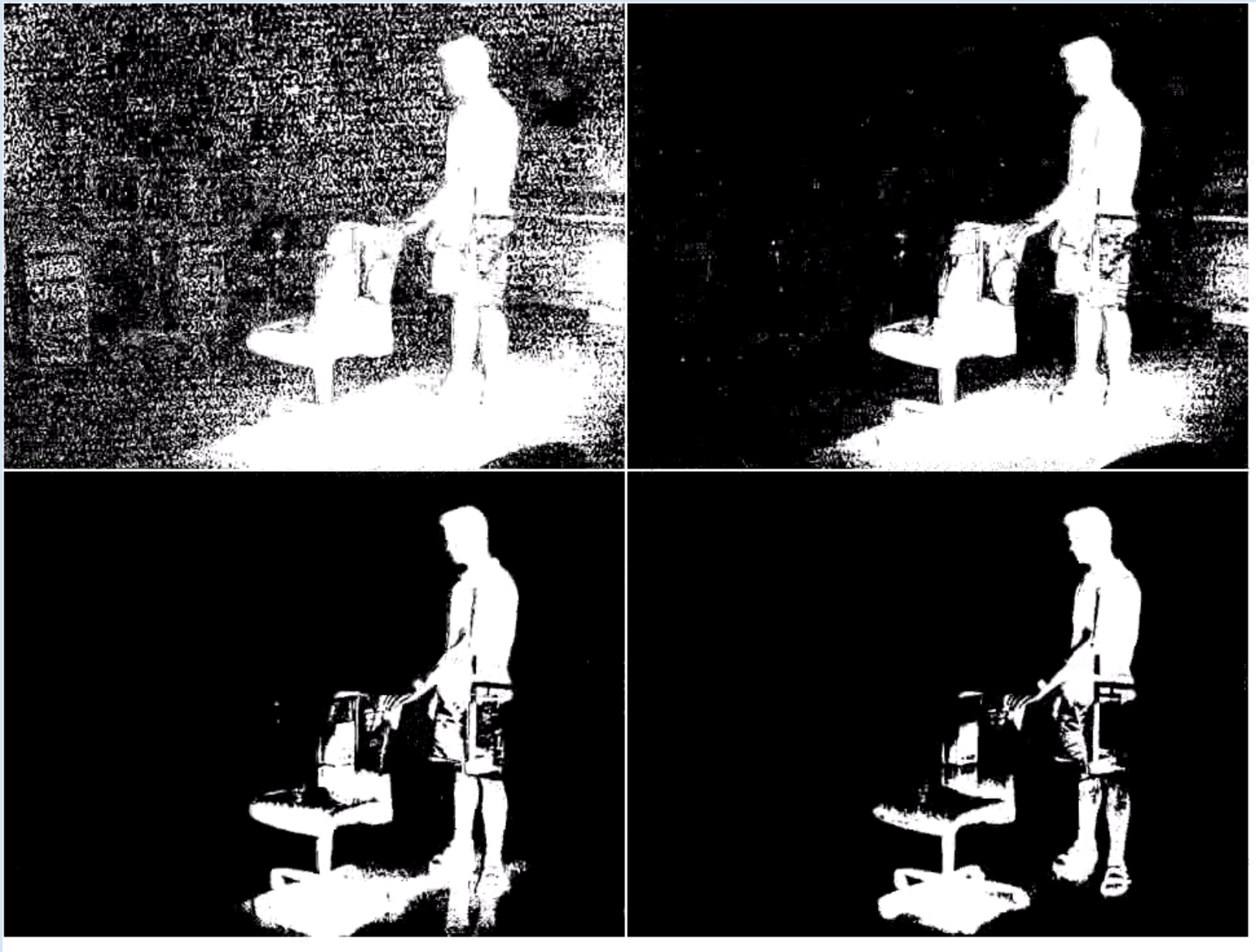
average of 20 images of empty scene

# Differences between background and actual images

video

# Problems with threshold





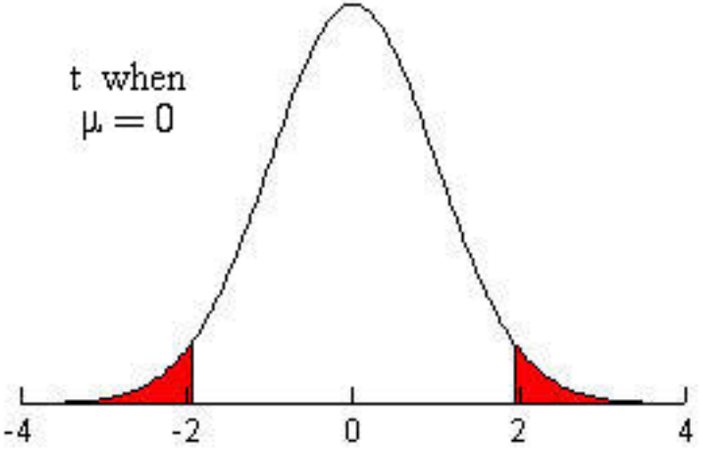
video

# Significance test with level $\alpha$

Take $n$ image frames of empty scene (hypothesis background $\mathcal{B}$), accumulate statistics on the $|I_t - B|$ measure (for all pixels together) and set $T$ such that

$$p(|I_t - B| > T \,|\, \mathcal{B}) = \alpha$$

e.g. significance level $\alpha = 0.01$ means $1\%$ of background pixels will be misclassified as foreground (so called false positives). Setting $\alpha$ too low increases foreground pixel misclassification (so called false negatives).



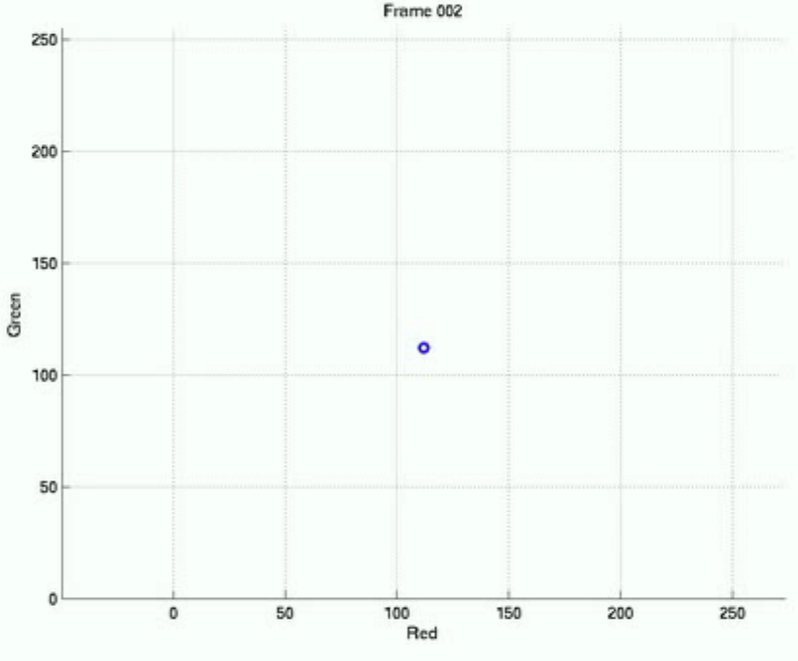
Accessible reading about Bayesian decision theory [1, Chapter 2]

# Better?

video

# Problems with the static models

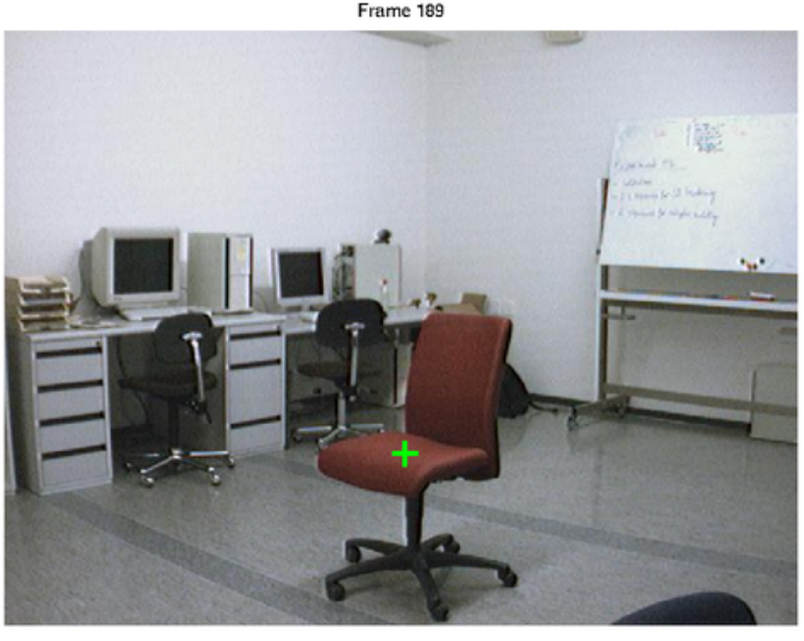
- number of missclassified pixels known only on the training data
- strictly require empty scene for learning (consider outdoor surveillance)
- do not adapt to changing environment

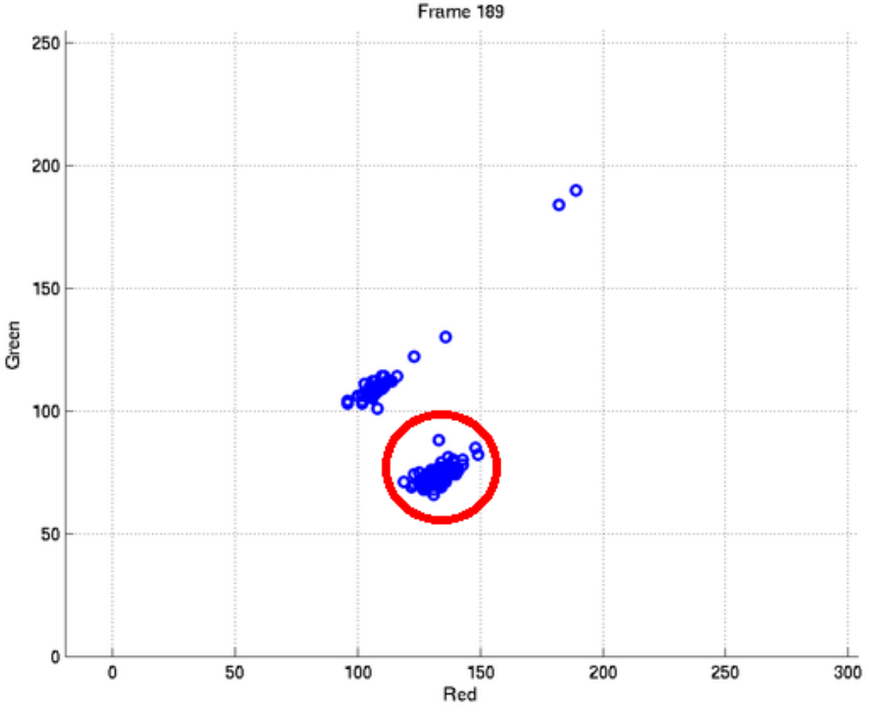



video

# Illumination — dim

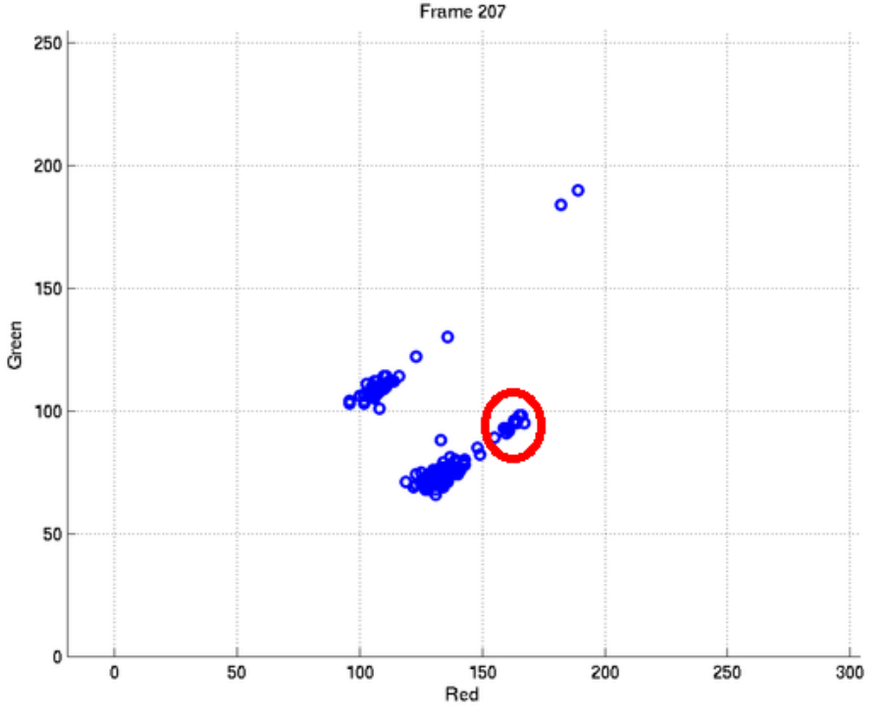
Observation: changing illumination (including shadows cast by moving objects) has multiplicative effect on the observed intensity.



# Illumination — more light

Observation: changing illumination (including shadows cast by moving objects) has multiplicative effect on the observed intensity.



# Illumination-invariant color spaces

There are different ways to exploit this observation:

- transform $[R, G, B]$ pixel color into color representation invariant to illumination intensity, e.g. to ratios $\left[\frac{R}{B}, \frac{G}{B}\right]$, $\frac{[R,G,B]}{\sqrt{R^2+G^2+B^2}}$ or opponent colors $[I, R-G, 2B-R-G]$
- examine gray values at neighbor pixels to find out if they have undergone the same transformation which could be explained by illumination change [2]

# Illumination-invariant color spaces

There are different ways to exploit this observation:

- transform $[R, G, B]$ pixel color into color representation invariant to illumination intensity, e.g. to ratios $\left[\frac{R}{B}, \frac{G}{B}\right]$, $\frac{[R,G,B]}{\sqrt{R^2+G^2+B^2}}$ or opponent colors $[I, R-G, 2B-R-G]$
- examine gray values at neighbor pixels to find out if they have undergone the same transformation which could be explained by illumination change [2]

**Problems**

- Dark areas $[R, G, B] \to 0$ (consider the fractions)
- Colors are not color enough (real world is grey, intensity is the dominant value)

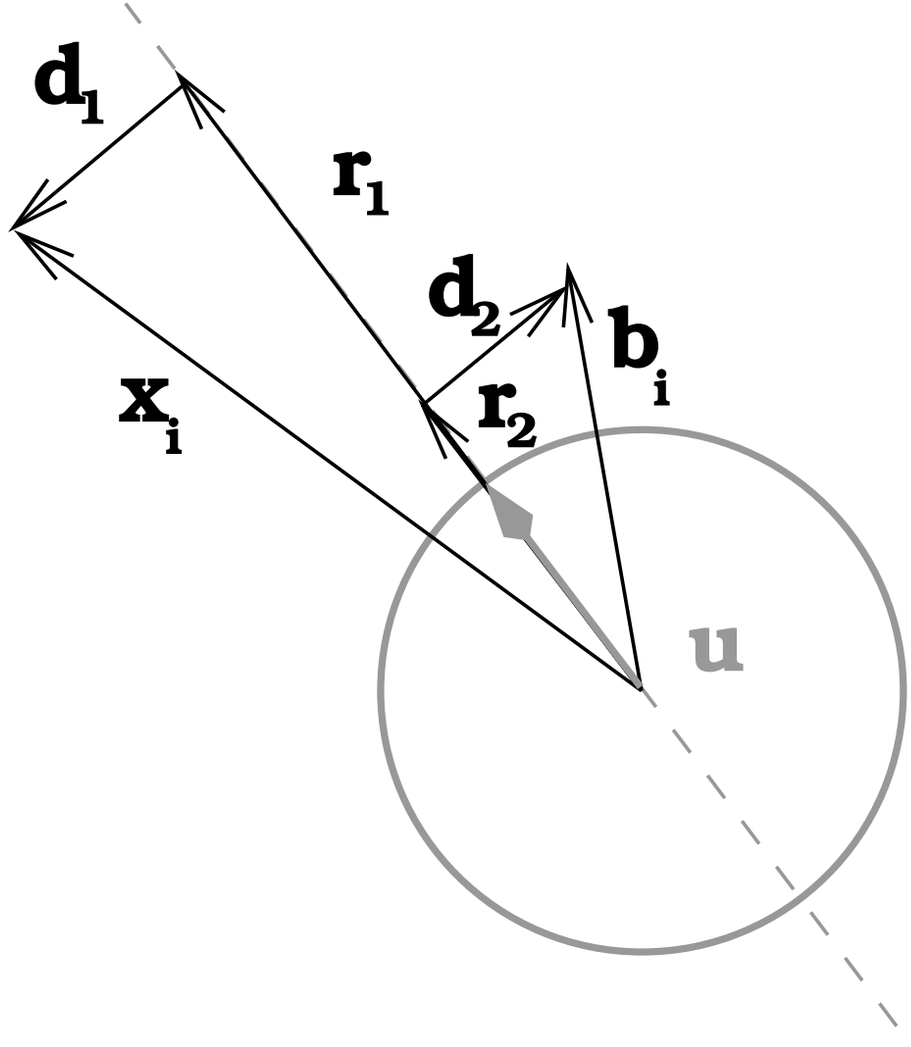
Still we can do something . . .

# Collinearity measurement

Consider not only pixel but also its neighbourhood

- Background: $\mathbf{b} = \mathbf{s} + \boldsymbol{\epsilon}_b$
- Observed signal: $\mathbf{x} = k\mathbf{s} + \boldsymbol{\epsilon}_x$
- $\mathbf{s}$ is the true (unknown) signal
- $k$ is a illumination factor
- $\boldsymbol{\epsilon}$ noise

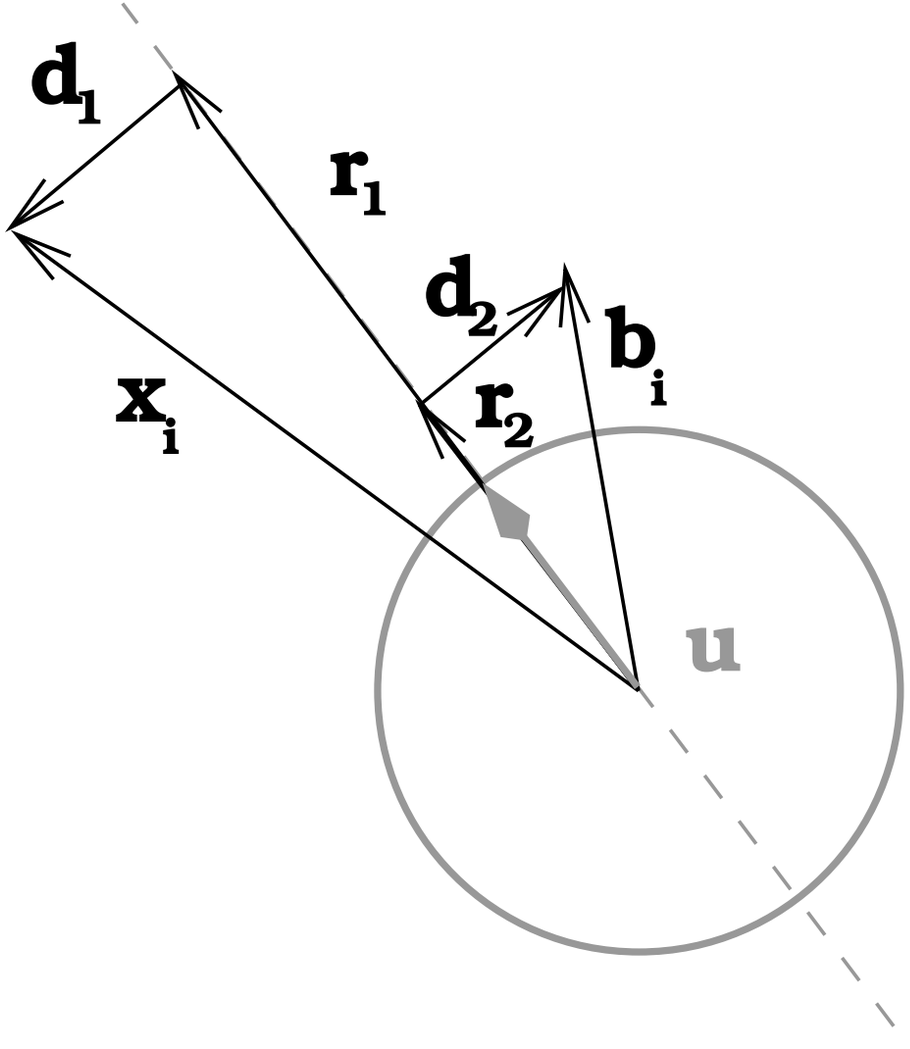
Change detection: Can be the $\mathbf{b}$ and $\mathbf{x}$ considered as collinear?



# Collinearity test statistics

Maximum likelihood (ML) estimate of the true signal direction $\mathbf{u}$ is given by minimizing the sum:

$$D^2 = \|\mathbf{d_1}\|^2 + \|\mathbf{d_2}\|^2$$

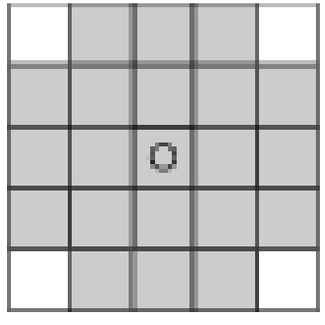


# Possible post-processing by using binary morphology



$X$

$B$



$X \ominus B$

$(X \ominus B) \oplus B$

# Adaptive background

Idea: the background model is learning from incoming image frames.

+ can be used in more general situations, also outdoors
– finding balance between learning background changes quickly and forgetting temporarily static objects
– does not distinguish between different objects

Simple modification of static background model: update the background model with learning rate $\alpha \in \langle 0, 1 \rangle$

$$B_t = (1 - \alpha) B_{t-1} + \alpha Y_t \ ,$$

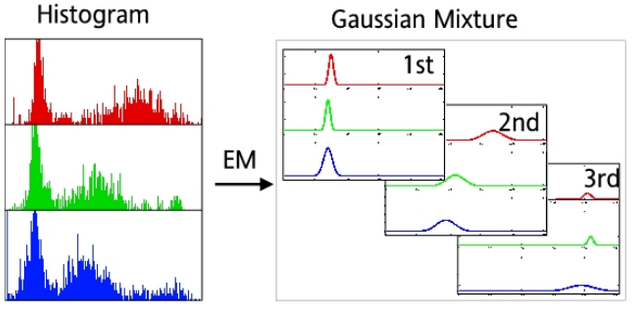
where $Y_t$ is current observation, may be pixel intensity, color, depth . . .

# Mixture of Gaussians

Observation: the changes in images of "static" background are due to:

- additive Gaussian noise
- multiple different surfaces (waving trees) or illuminations (sun/clouds)

Idea: model color of each pixel as a weighted set of $K$ 3-dimensional (red,green,blue) Gaussians. Each Gaussian has mean value in each dimension $\boldsymbol{\mu} = [\mu_R, \mu_G, \mu_B]^\top$ and diagonal covariance matrix $\Sigma = \mathrm{diag}(\sigma_R^2, \sigma_G^2, \sigma_B^2)$, i.e. the color channels are for simplicity assumed independent.




Used in different variations, good reference [3]

# Mixture of Gaussians 2

The probability of observing $\mathbf{y}_t$ according to the Gaussian mixture model of background $\mathcal{B}$ is then:

$$p_{\mathcal{B}}(\mathbf{y}_t \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_{t-1}) = \sum_{k=1}^{K} w_{t-1,k}\ \eta(\mathbf{y}_t,\ \boldsymbol{\mu}_{t-1,k},\ \Sigma_{t-1,k}) \quad ,$$

where $w_{t,k} \in \langle 0, 1 \rangle$ is weight of $k$-th Gaussian and $\eta$ is probability density function (standard Gaussian distribution)

$$\eta(\mathbf{y}, \boldsymbol{\mu}, \Sigma) = \frac{1}{\sqrt{(2\pi)^{d/2}|\Sigma|}}\ e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^{\top}\Sigma^{-1}(\mathbf{y}-\boldsymbol{\mu})} \quad .$$

To label the pixel as background, probability of the observation $\mathbf{y}_t$ has to be over certain threshold $p_{\mathcal{B}}(\mathbf{y}_t) > threshold$ .

# Updating Gaussian

Update the Gaussians for which $\eta(\mathbf{y}_t, \ldots) > threshold$ (i.e. those matching the observation $\mathbf{y}_t$) using an on-line K-means approximation

$$\boldsymbol{\mu}_{t,k} = (1-\alpha)\boldsymbol{\mu}_{t-1,k} + \alpha \mathbf{y}_t$$

$$\begin{aligned}
\sigma^2_{R,t,k} &= (1-\alpha)\sigma^2_{R,t-1,k} + \alpha(y_{R,t} - \mu_{R,t-1,k})^2 \\
\sigma^2_{G,t,k} &= (1-\alpha)\sigma^2_{G,t-1,k} + \alpha(y_{G,t} - \mu_{G,t-1,k})^2 \\
\sigma^2_{B,t,k} &= (1-\alpha)\sigma^2_{B,t-1,k} + \alpha(y_{B,t} - \mu_{B,t-1,k})^2
\end{aligned}$$

More precise way of update would be using Expectation-Maximization on a number of recent observations, it is however more time consuming.

# Updating weights of Gaussians in mixture

The weight of all the Gaussians in mixture is decreased

$$w_{t,k} = (1 - \alpha) w_{t-1,k}$$

with the exception of the Gaussians that were updated, for which

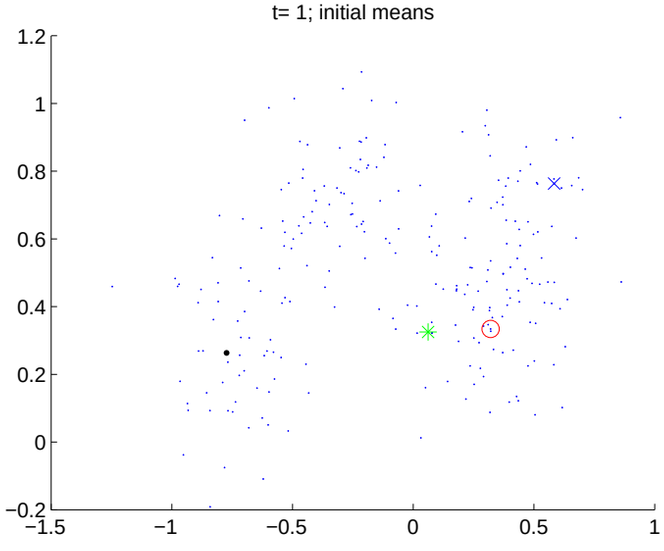
$$w_{t,k} = (1 - \alpha) w_{t-1,k} + \alpha \ .$$

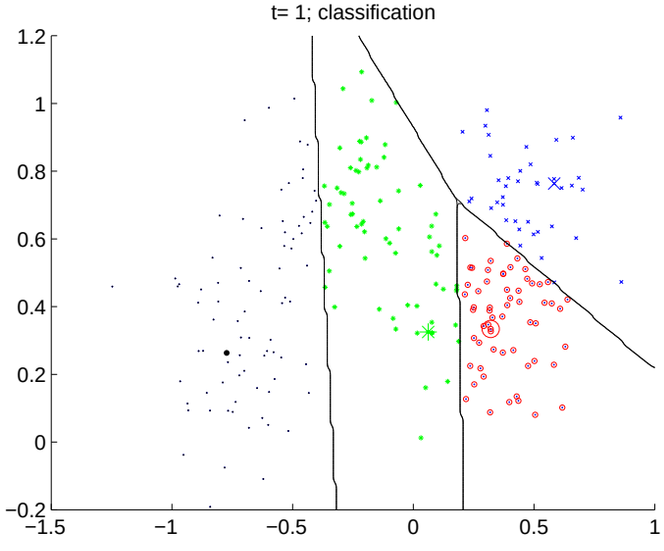
If $\mathbf{y}_t$ matched no Gaussian, the lowest-weight Gaussian from mixture is removed and new one is added with mean $\boldsymbol{\mu} = \mathbf{y}_t$, variances for all channels $\sigma^2 = \sigma^2_{init}$ and $w = \alpha$.

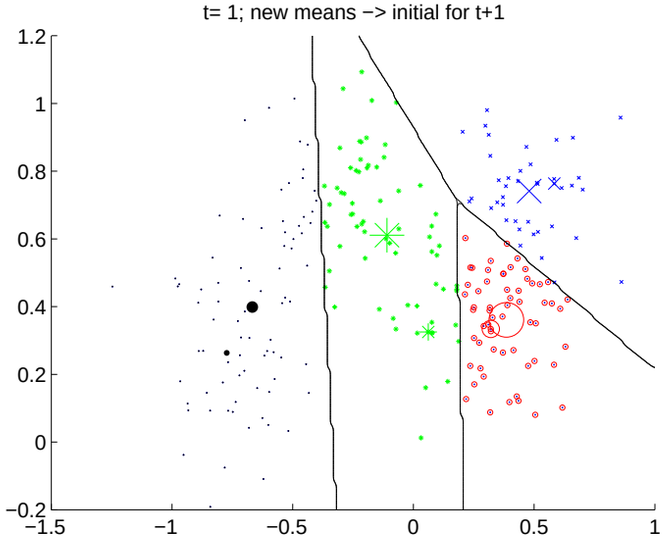
Weights are normalized so that $\sum_{k=1}^{K} w_{t,k} = 1$.

# How the K-means works, step 1



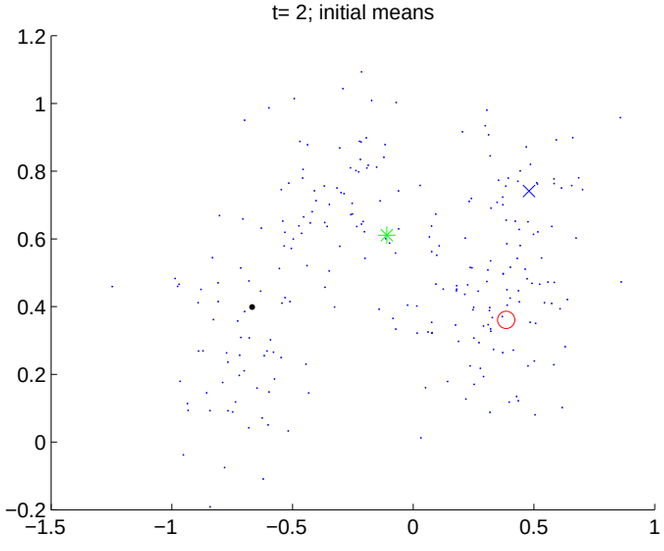


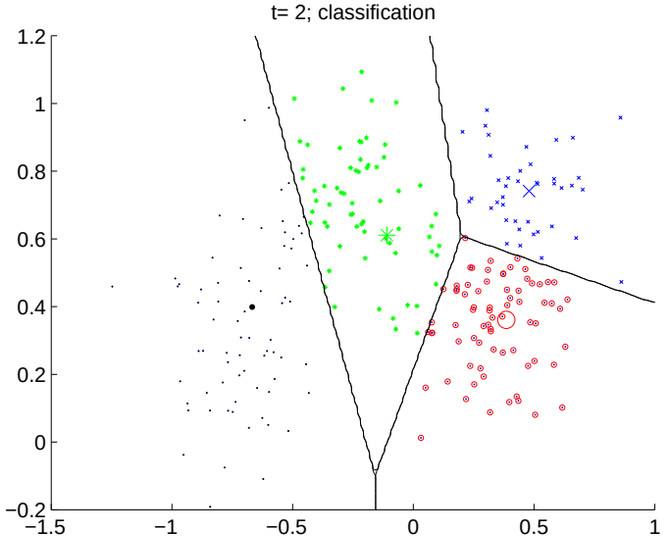



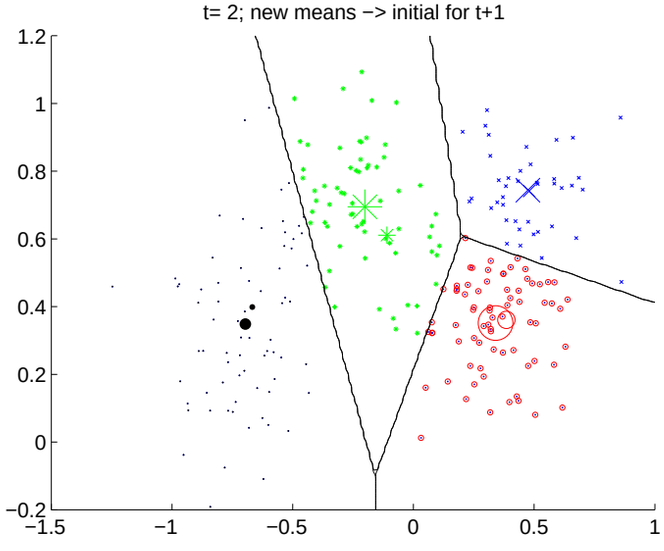



# How the K-means works, step 2, . . .

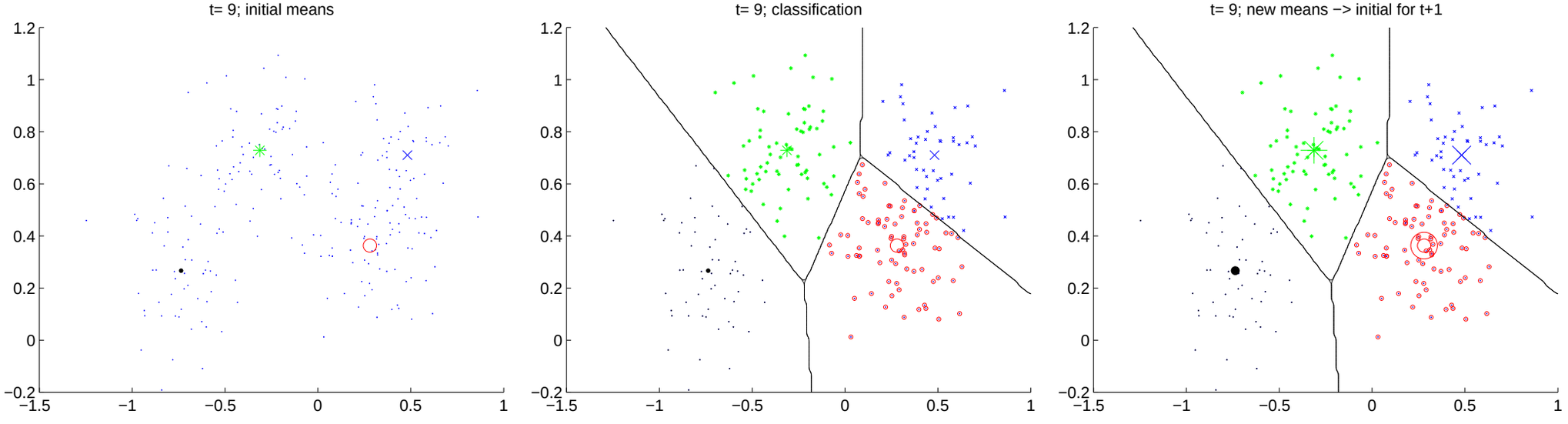




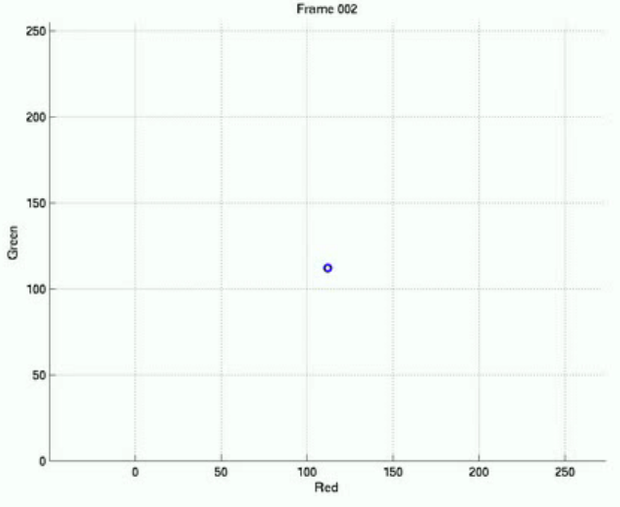



# . . . until it converges

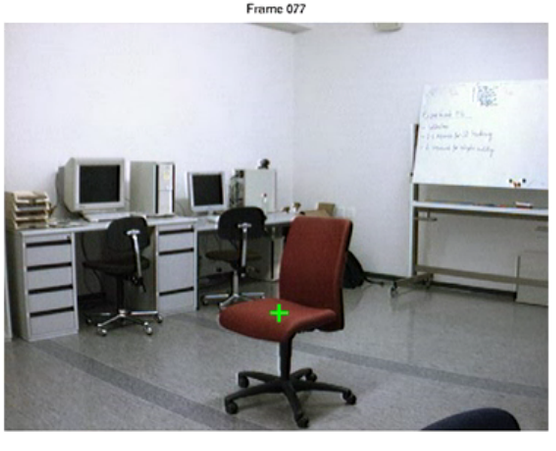


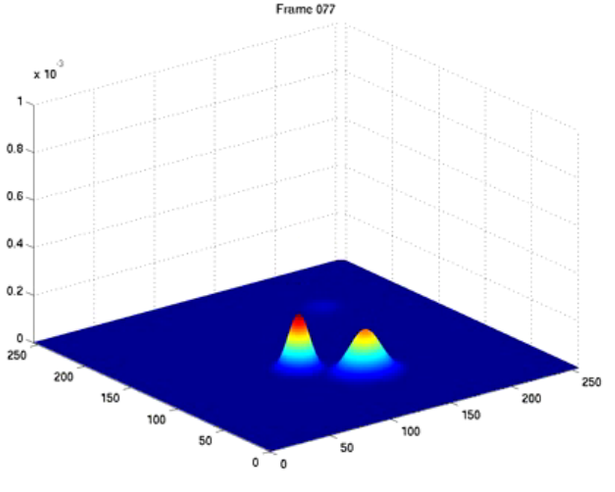

The iterations stop when there is no more change in the classification.

# Mixture update visualization



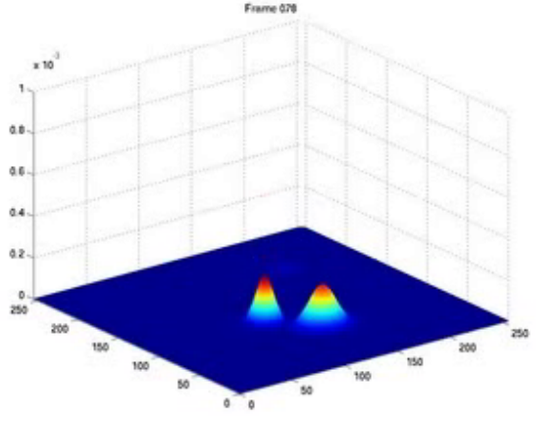

video

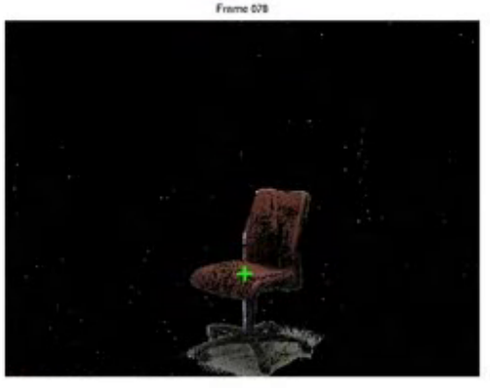



video

# Adaptive segmentation results




video

video

# Motion segmentation summary

- pixel based approach
- no general all-solving method
- despite limited power widely used because of its simplicity

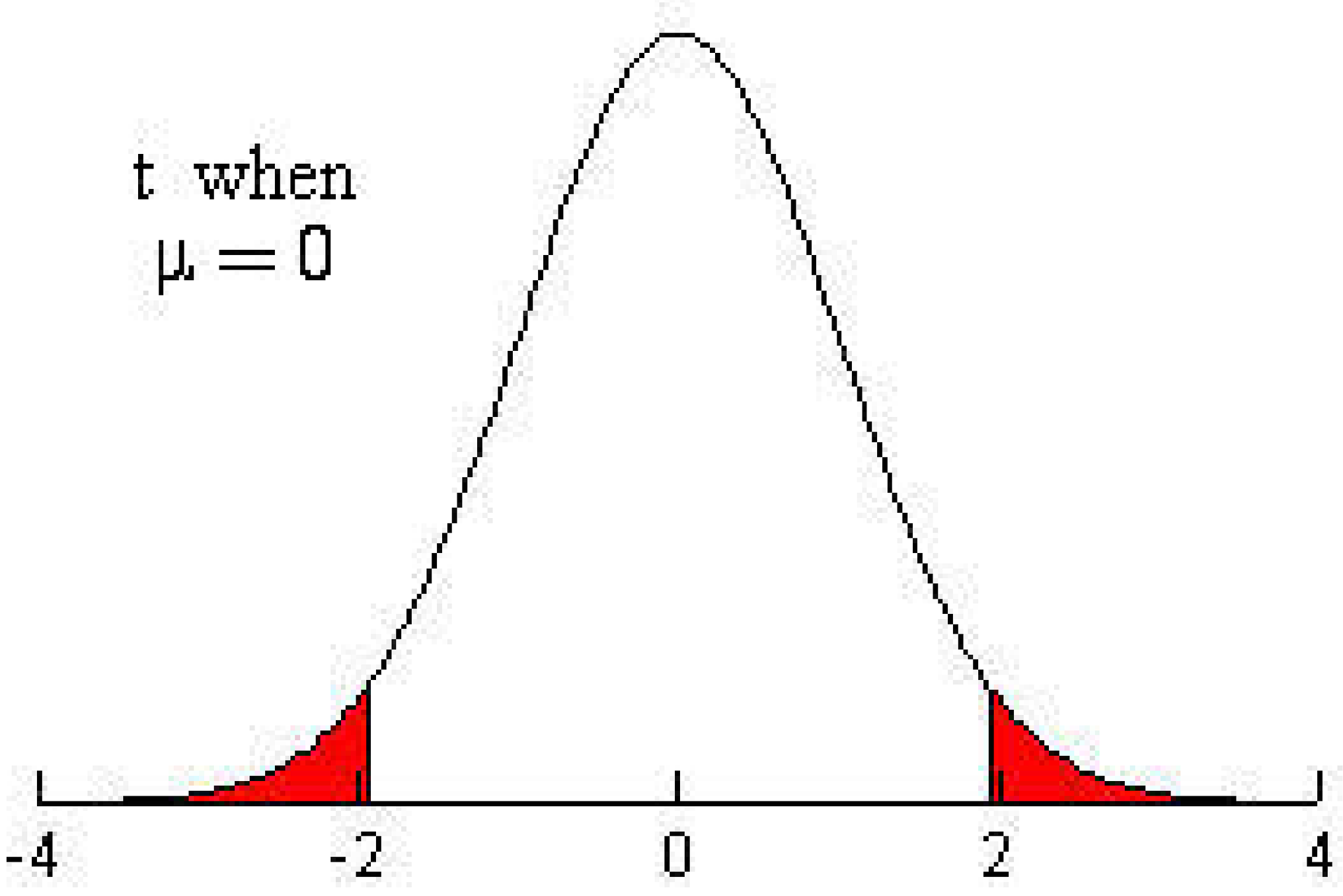
# References


[1] Richard O. Duda, Peter E. Hart, and David G. Stork. Pattern Classification. John Wiley & Sons, New York, NY, USA, 2nd edition, 2001.

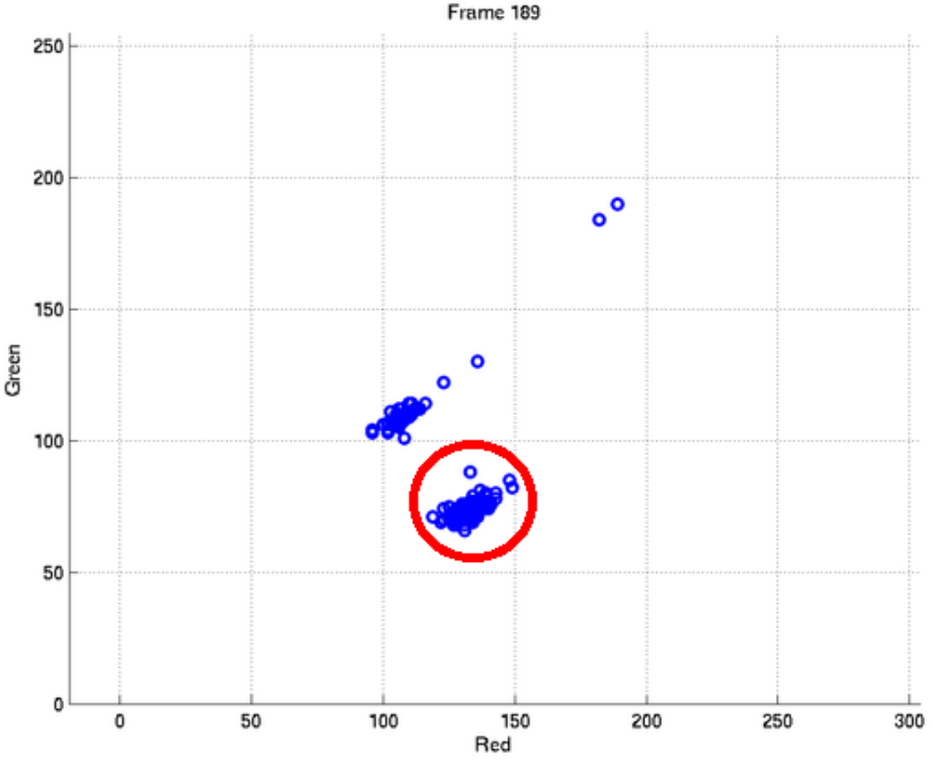
[2] Rudolf Mester, Til Aach, and Lutz Dümbgen. Illumination-invariant change detection using statistical colinearity criterion. In R. Radig and Florczyk S., editors, DAGM2001, number 2191 in LNCS, pages 170–177. Springer–Verlag, 2001.

[3] Chris Stauffer and W.E.L Grimson. Adaptive background mixture models for real-time tracking. In IEEE Computer Vision and Pattern Recognition, volume 2, pages 2242–2252, June 1999.

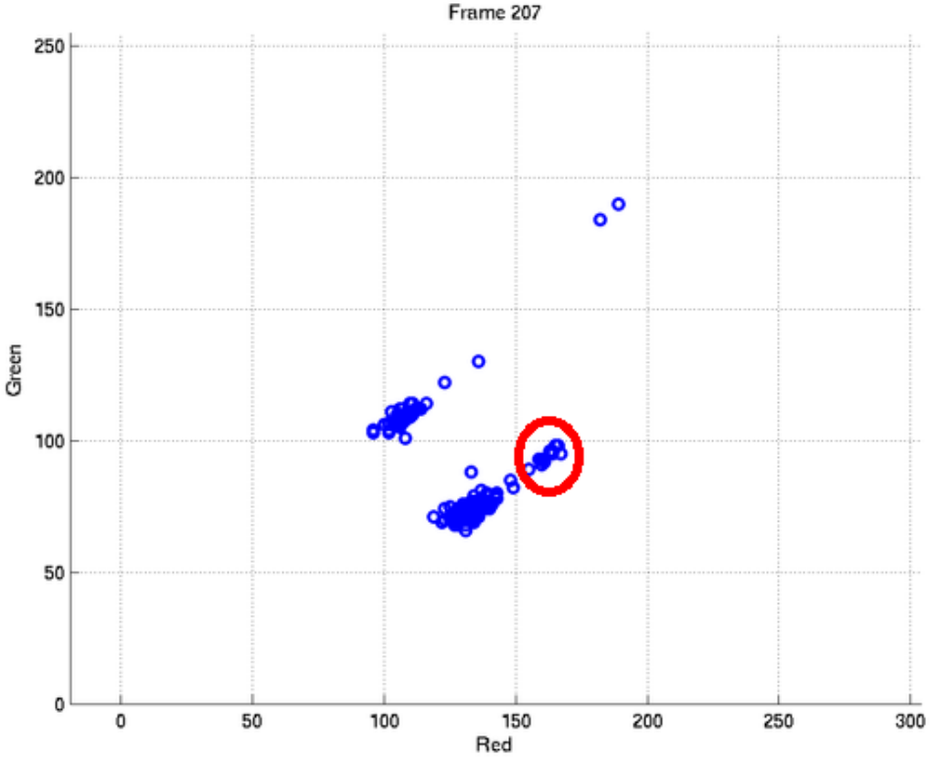
# End

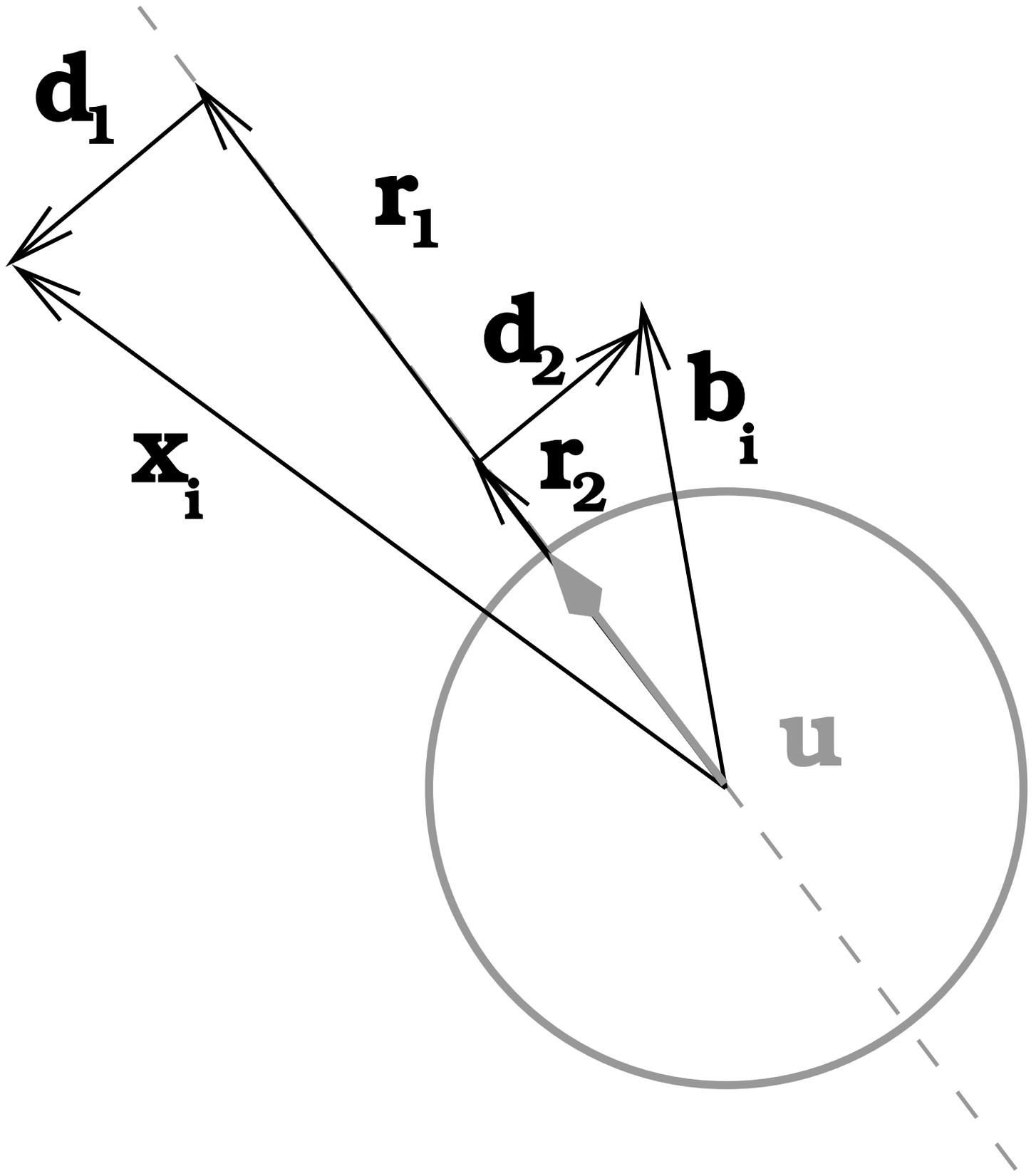

t when
$\mu = 0$
-4
-2
0
2
4

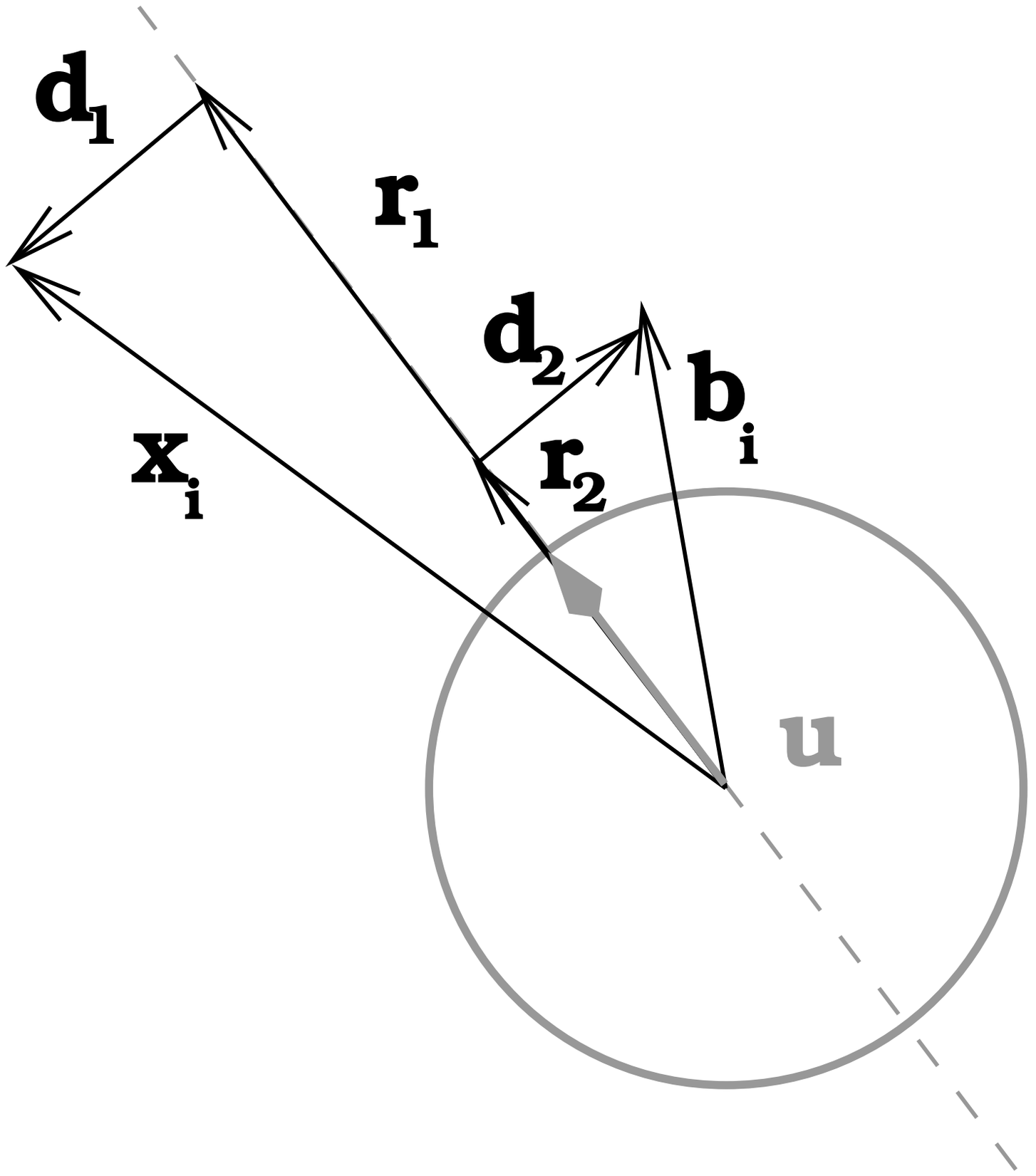
Frame 189


Frame 189
Green
Red
0
50
100
150
200
250
300

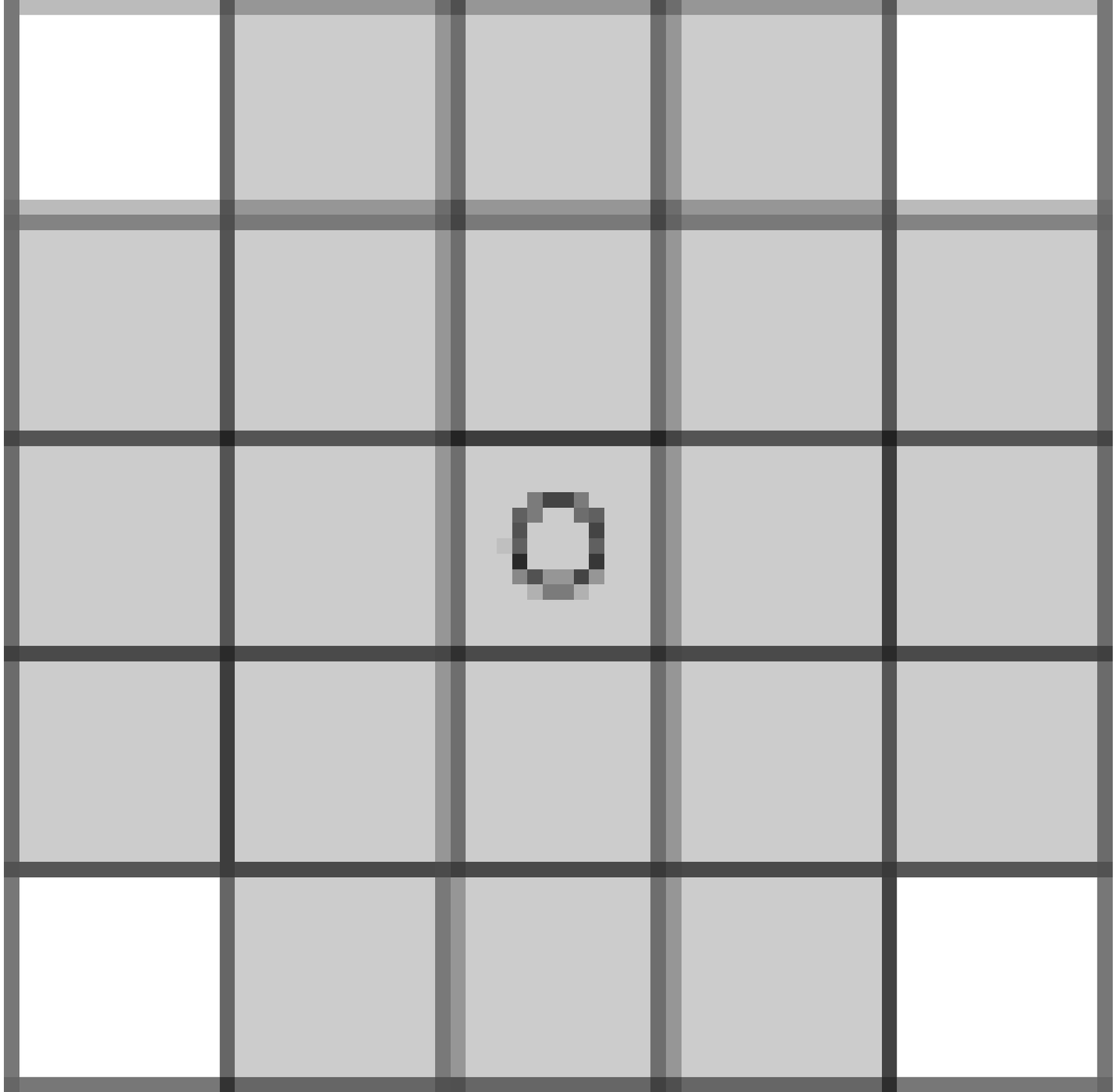
Frame 207


Frame 207
Green
Red
0
50
100
150
200
250
300

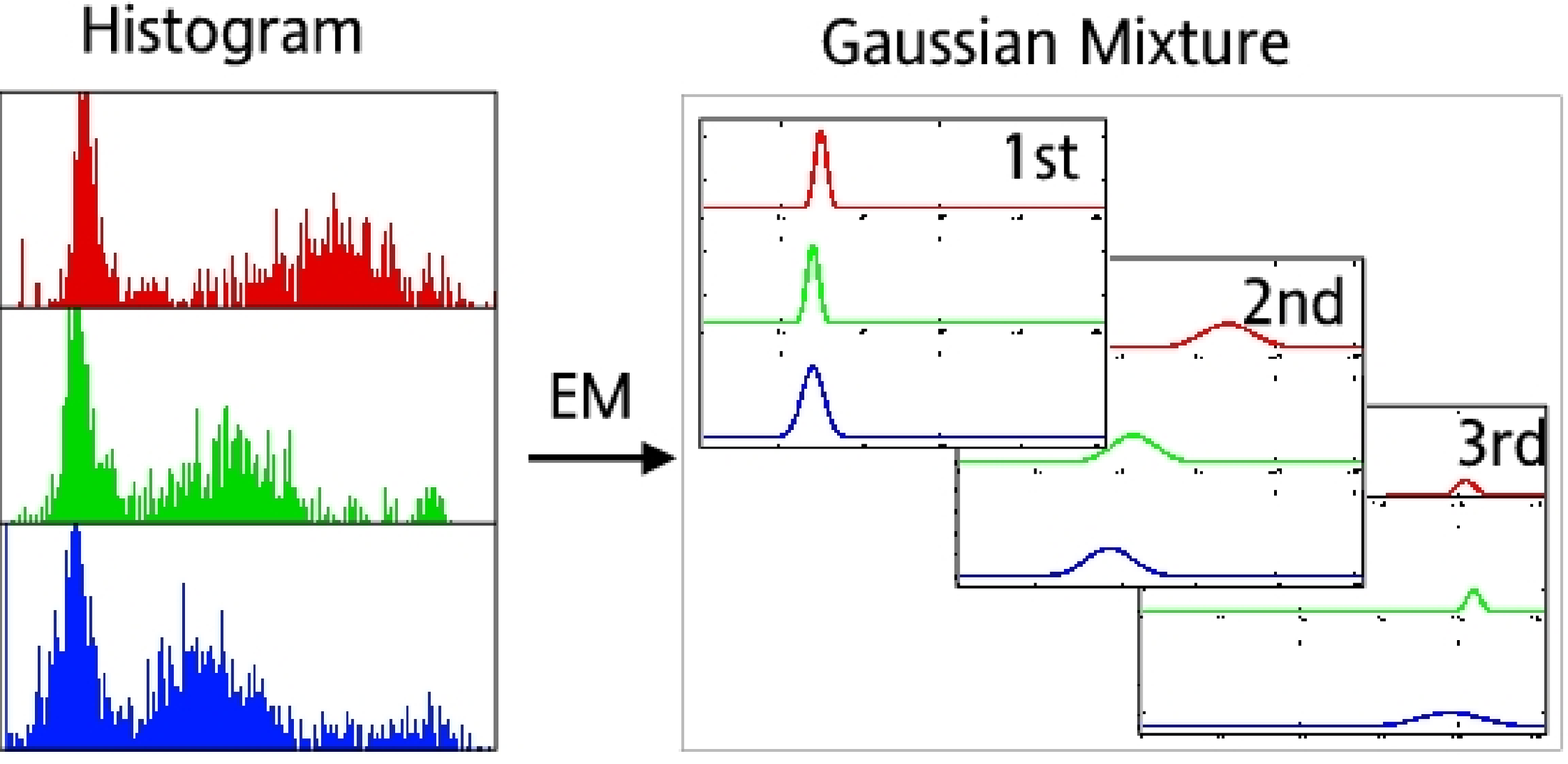

$d_1$
$r_1$
$d_2$
$b_i$
$x_i$
$r_2$
u

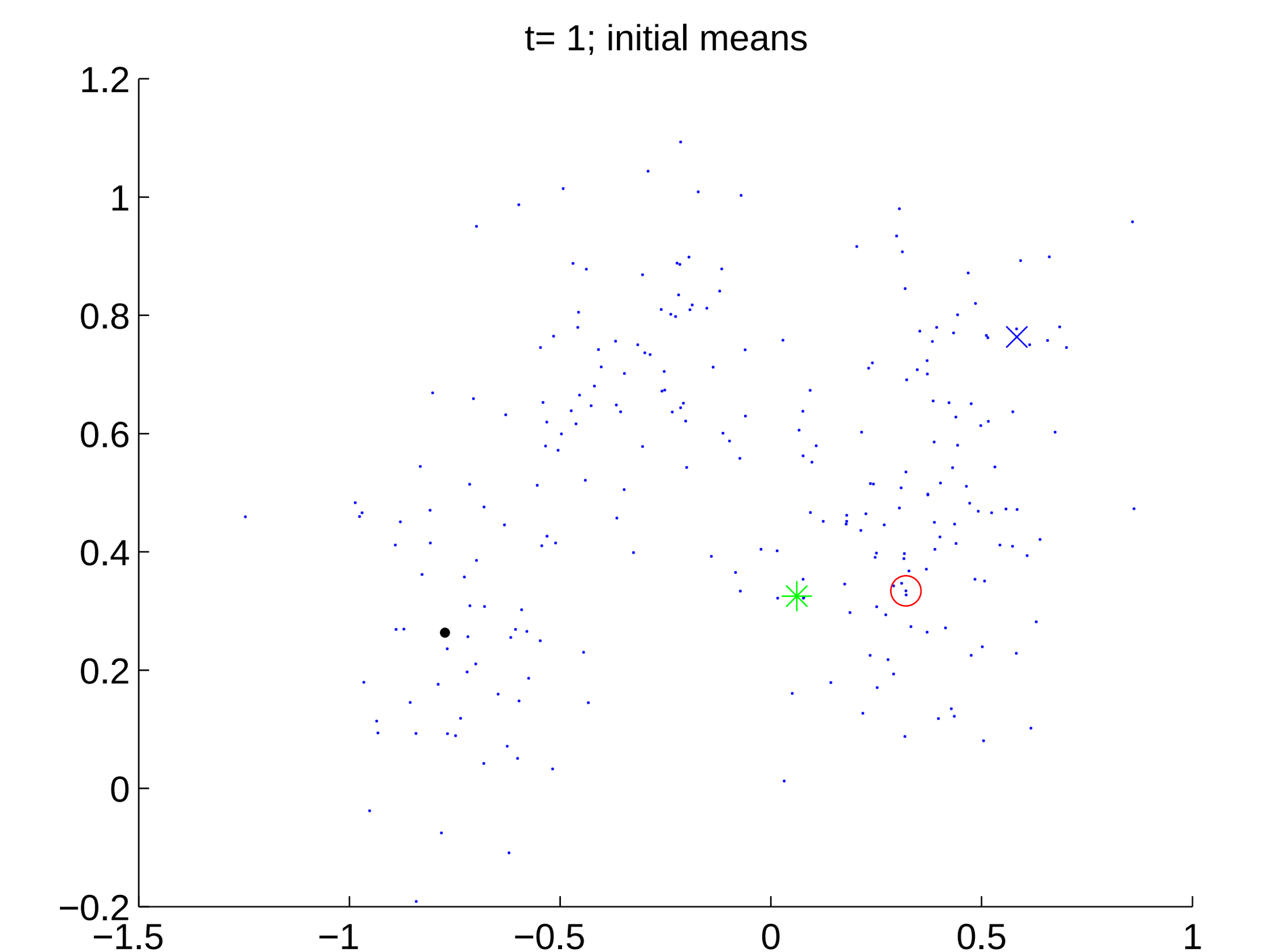

$d_1$
$r_1$
$d_2$
$b_i$
$x_i$
$r_2$
u

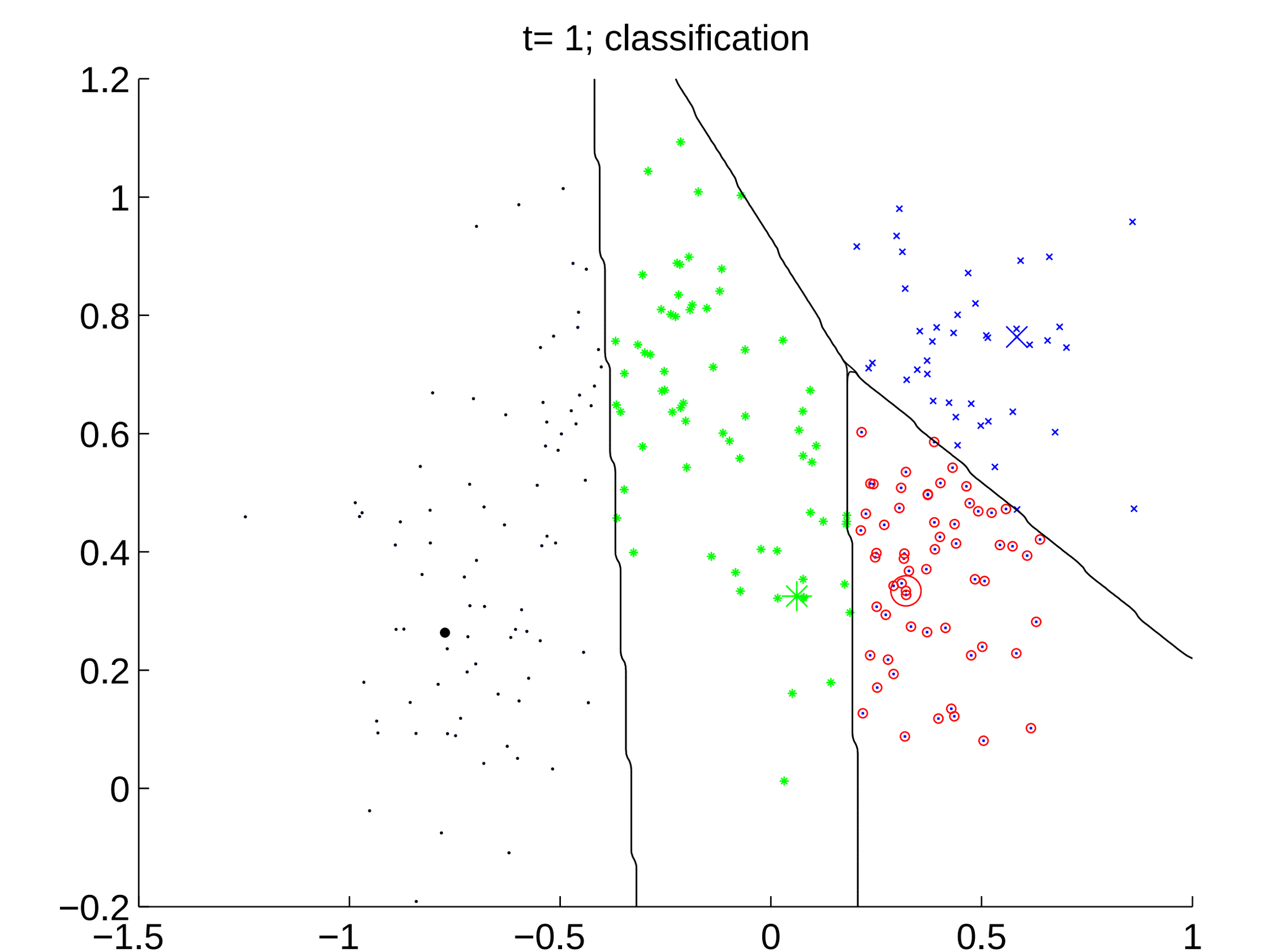
Floor...

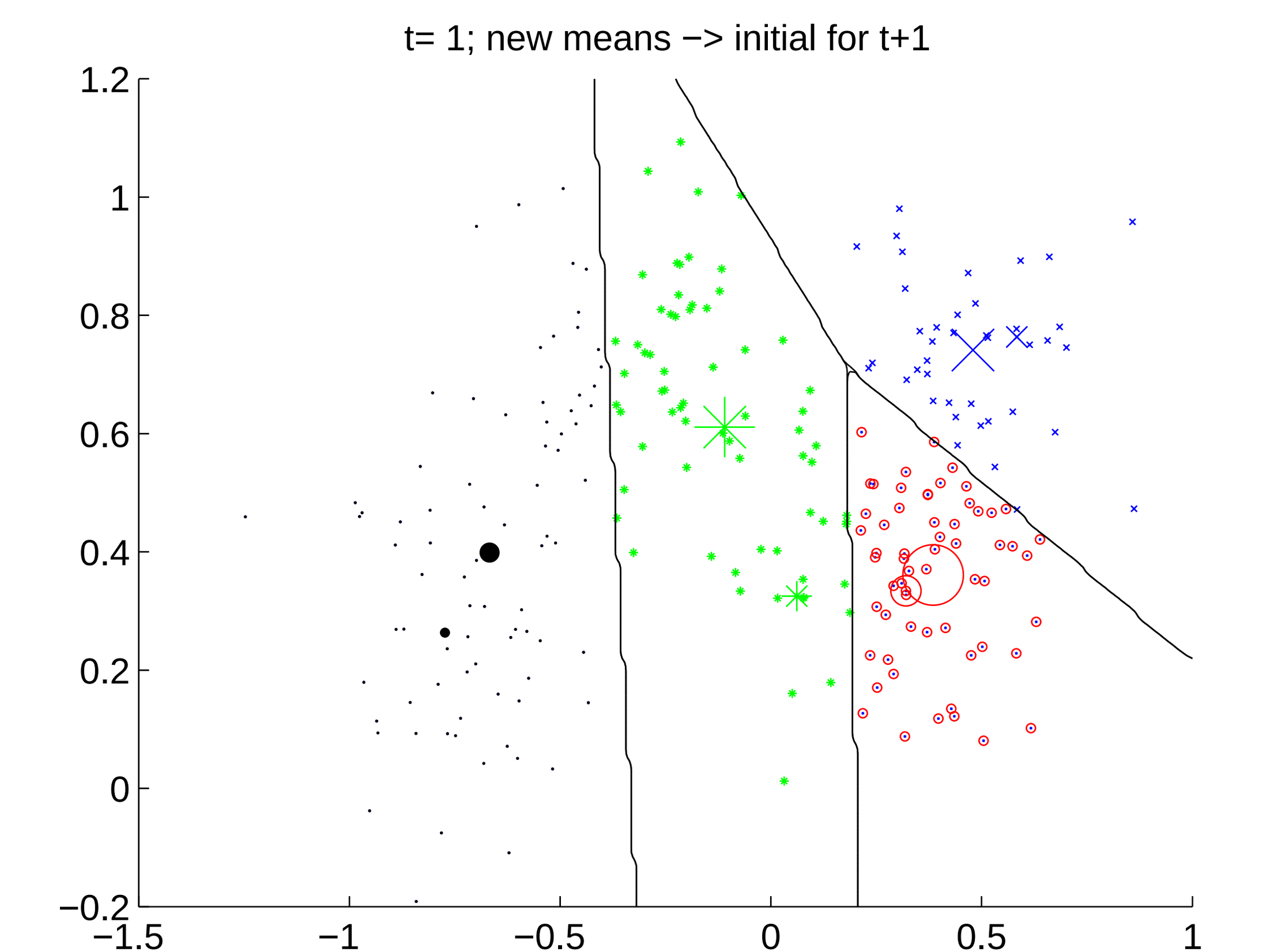


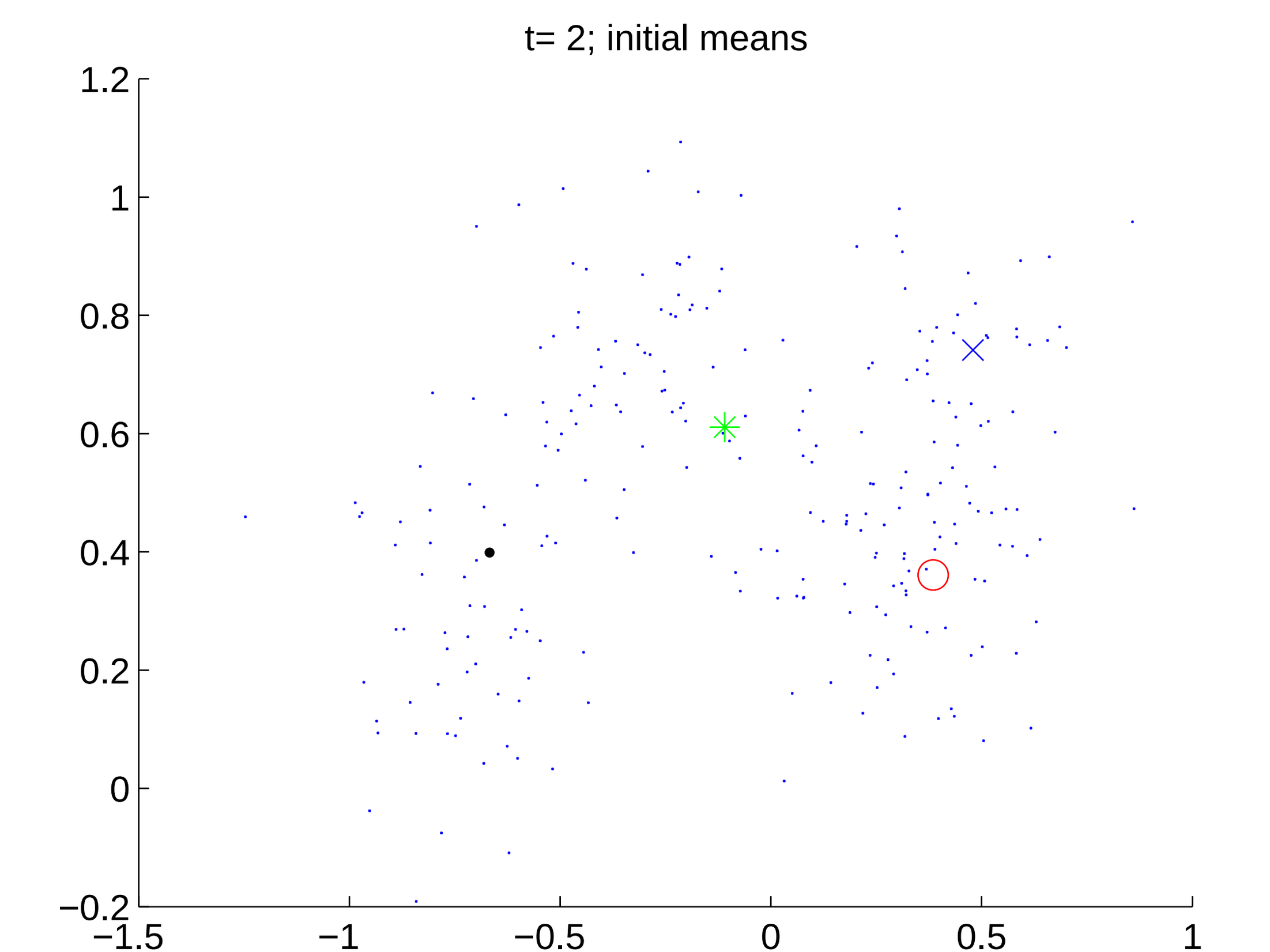
Floor

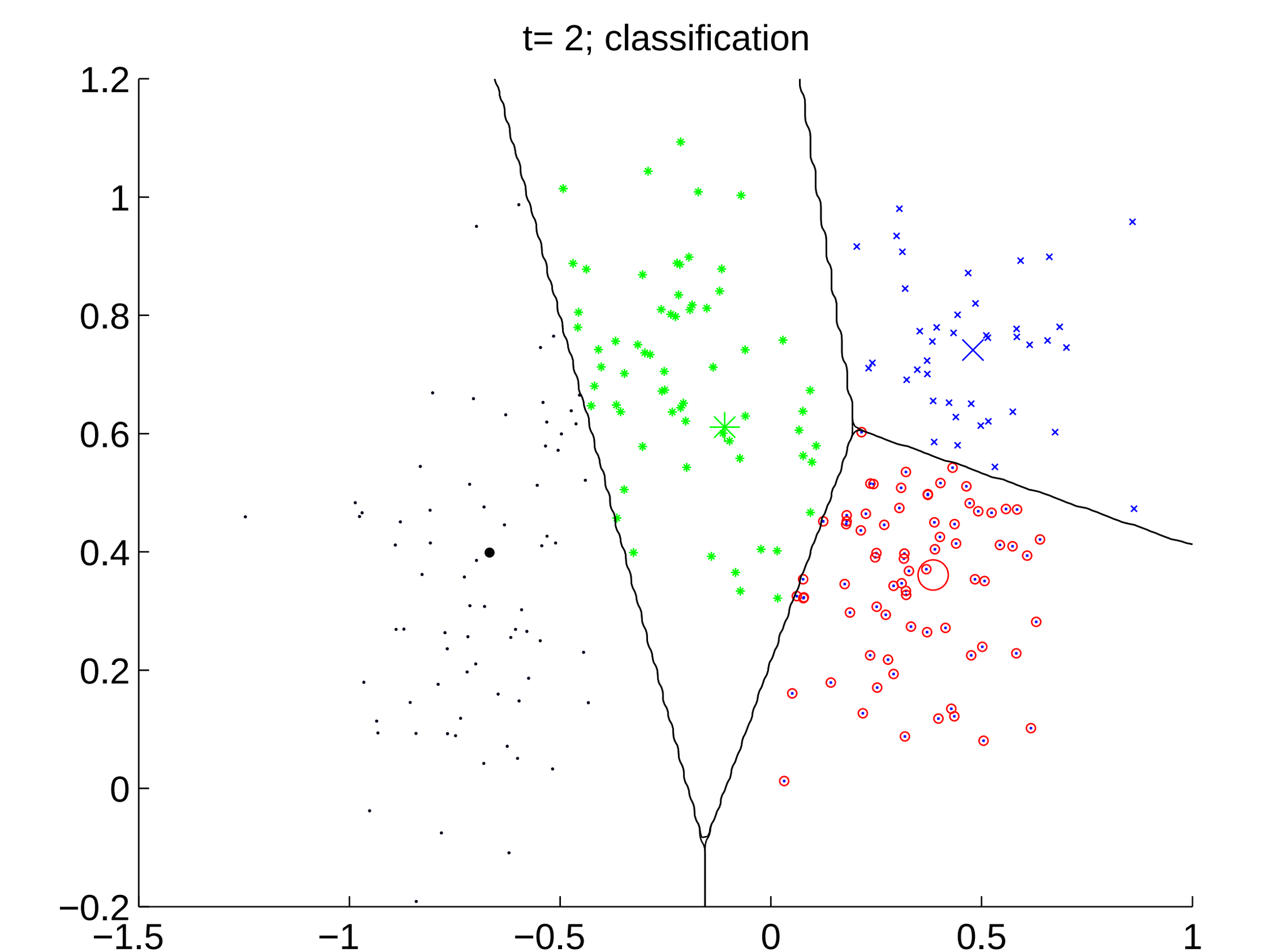

Histogram
EM
Gaussian Mixture
1st
2nd
3rd

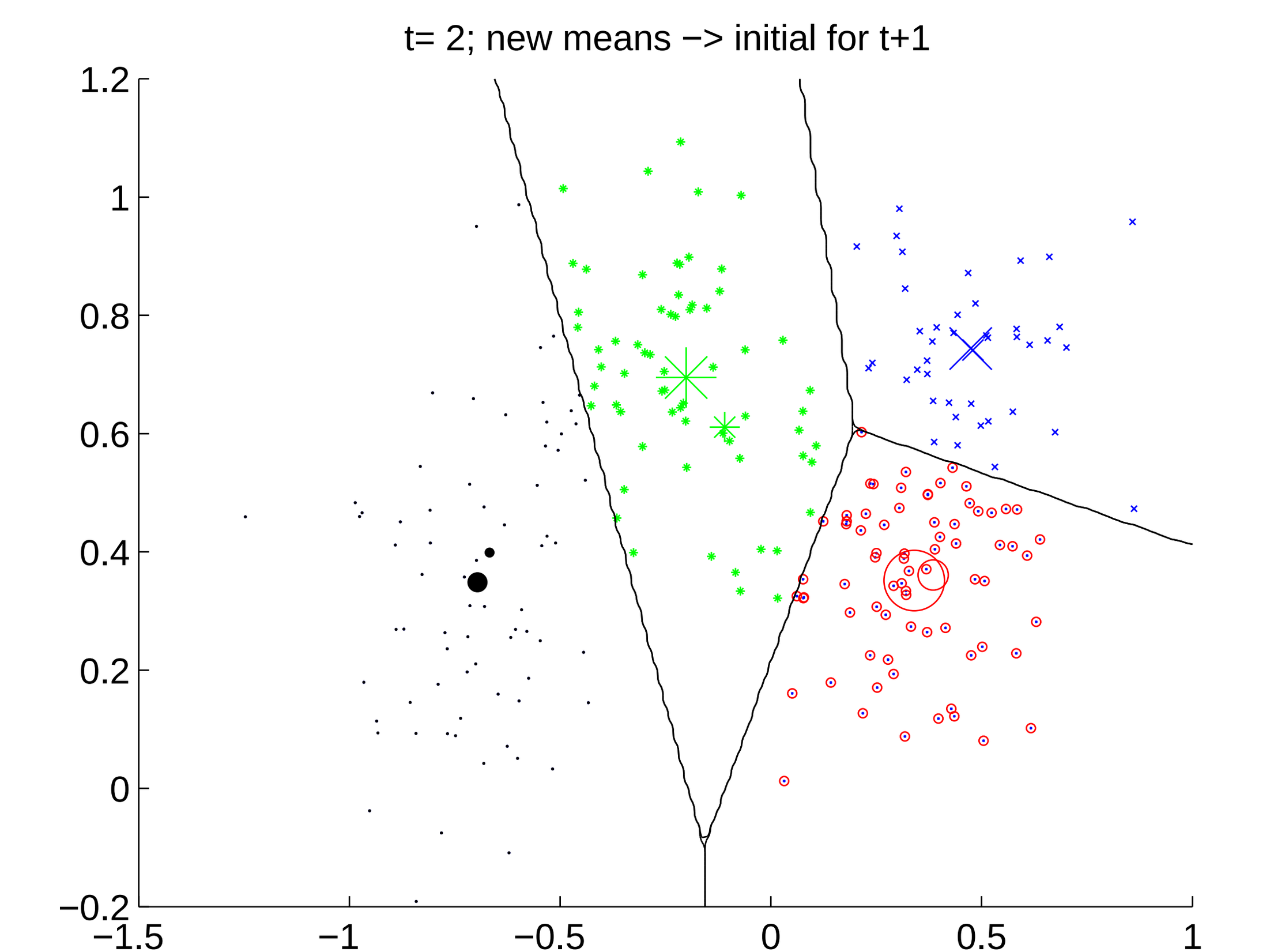

t= 1; initial means
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1

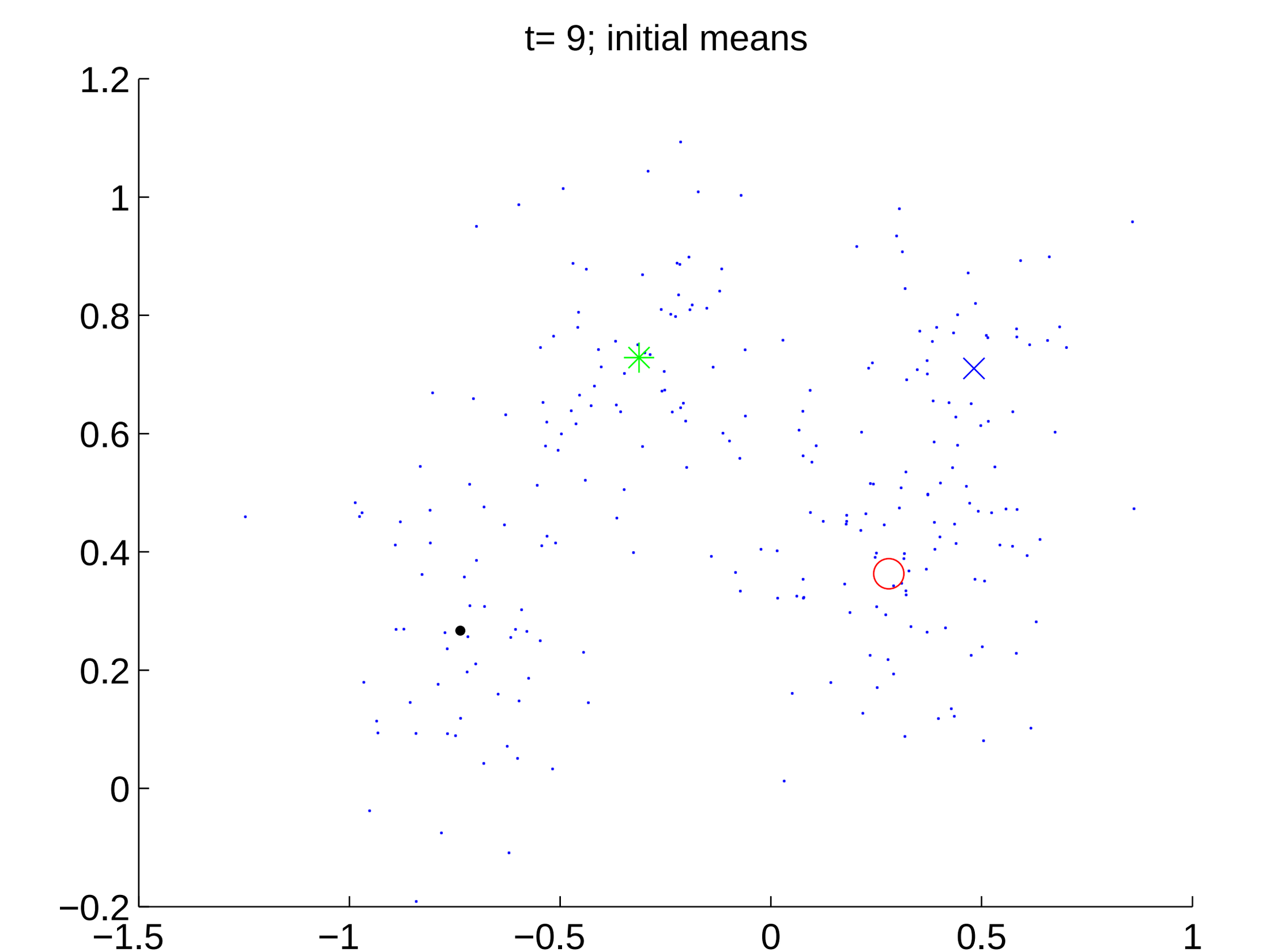

t= 1; classification
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1

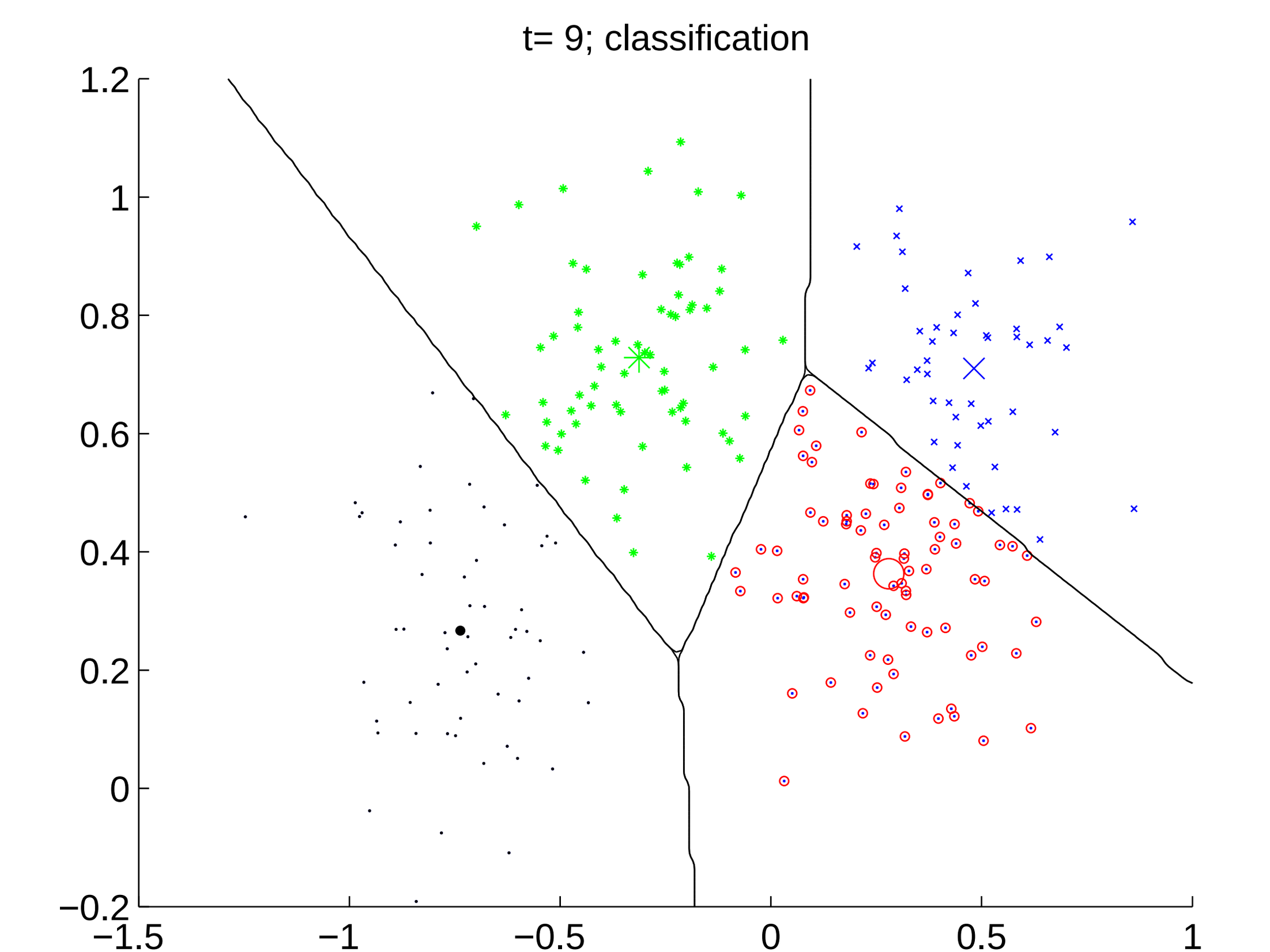

t= 1; new means −> initial for t+1
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1

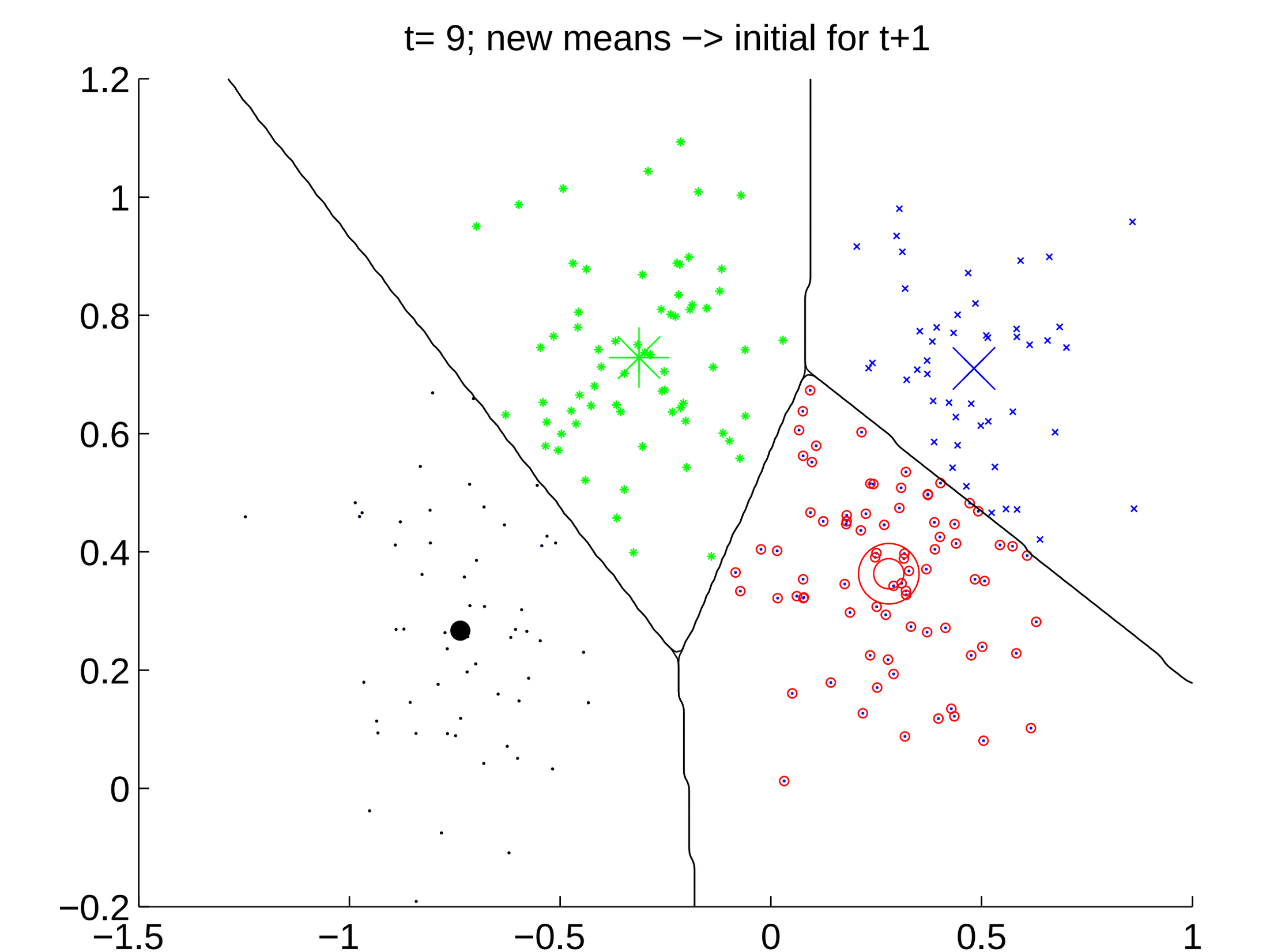

t= 2; initial means
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1


t= 2; classification
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1


t= 2; new means −> initial for t+1
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1


t= 9; initial means
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1


t= 9; classification
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1


t= 9; new means −> initial for t+1
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1.5
−1
−0.5
0
0.5
1