Univerza v Ljubljani

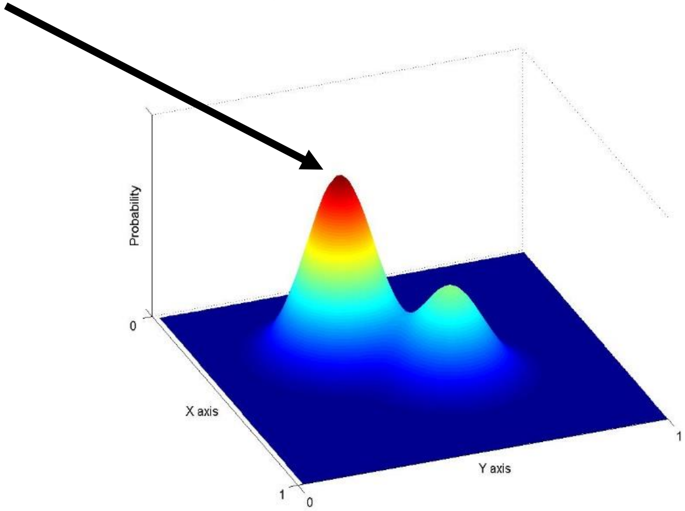
Mean-shift

Advanced Computer Vision Methods
Exercise 2

Visual Cognitive Systems Laboratory,
Faculty of Computer and Information Science,
University of Ljubljana

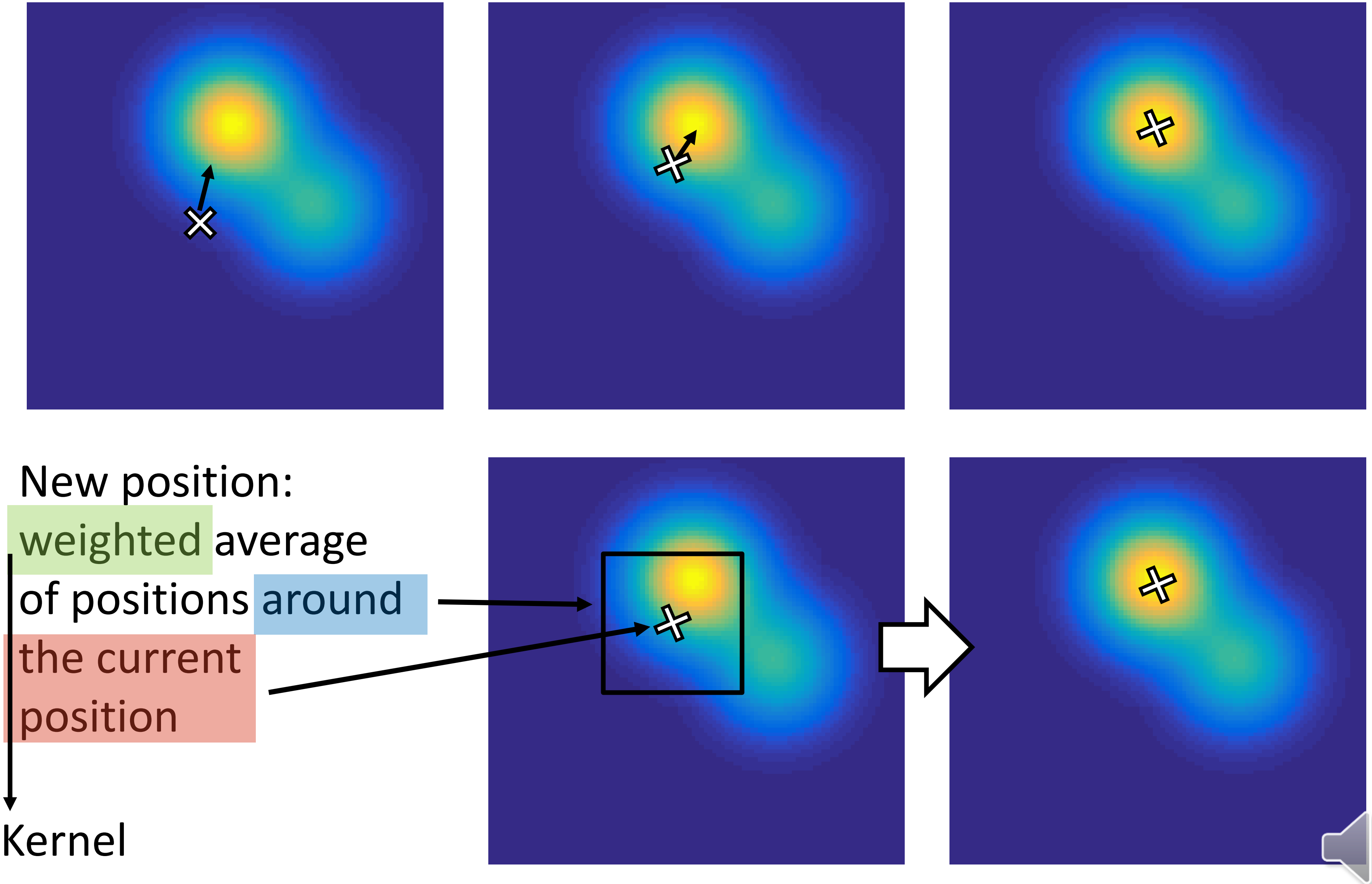
Mean-shift: idea

- Method for mode seeking in prob. density function



- We can calculate probability in each point and take the maximum
    - Slow approach
- We do not need to calculate probability in each point
- Start in some point and continue in the direction of a gradient

Mean-shift: iterative mode seeking



New position:
weighted average
of positions around
the current
position

Kernel

Mean-shift: equation

- Iteration to calculate new position (calculate twice: for X and Y direction)

$$x^{(k+1)} = \frac{\sum_{i=1}^{N} x_i w_i g\left(\left\|\frac{x^{(k)}-x_i}{h}\right\|^2\right)}{\sum_{i=1}^{N} w_i g\left(\left\|\frac{x^{(k)}-x_i}{h}\right\|^2\right)}$$

x_i represents coordinates within the window:

-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2

For X direction

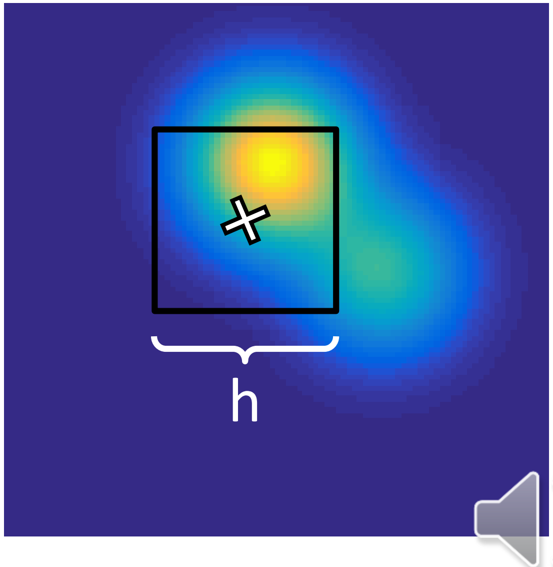
-2	-2	-2	-2	-2
-1	-1	-1	-1	-1
0	0	0	0	0
1	1	1	1	1
2	2	2	2	2

For Y direction

Kernel derivative: $g(x) = -k'(x)$

Function value in x_i: w_i

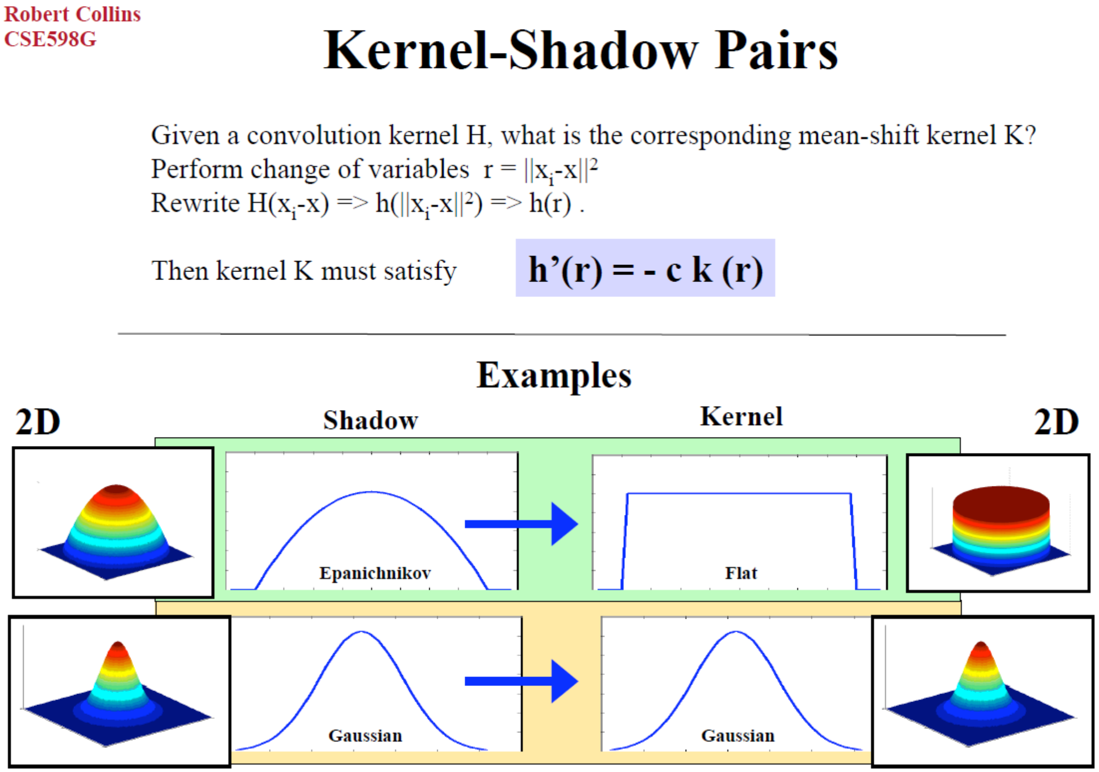
Bandwidth (size of kernel): h



Mean-shift: kernels

$$g(x) = -k'(x)$$

$$x^{(k+1)} = \frac{\sum_{i=1}^{N} x_i w_i g\left(\left\| \frac{x^{(k)} - x_i}{h} \right\|^2\right)}{\sum_{i=1}^{N} w_i g\left(\left\| \frac{x^{(k)} - x_i}{h} \right\|^2\right)}$$



Robert Collins
CSE598G

Kernel-Shadow Pairs

Given a convolution kernel H, what is the corresponding mean-shift kernel K?
Perform change of variables $r = ||x_i - x||^2$
Rewrite $H(x_i - x) \Rightarrow h(||x_i - x||^2) \Rightarrow h(r)$.

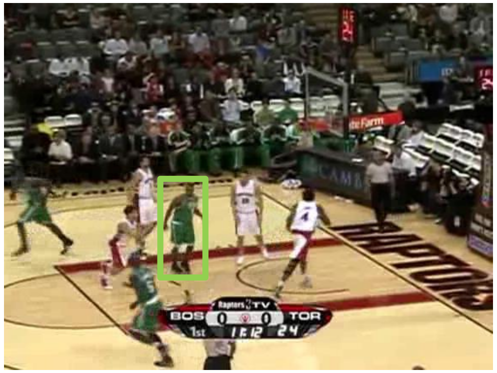
Then kernel K must satisfy **$h'(r) = - c\ k\ (r)$**

Examples

Biweight → Epanechnikov

Tracker design

Frame 1



Frame 2

…

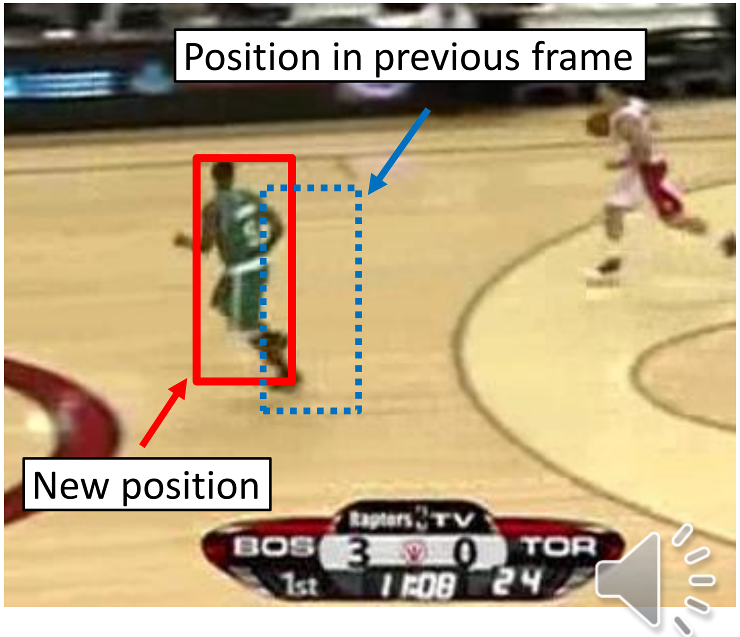
Frame t

…

Given ground-truth bounding box

Tracking algorithm:
initialize tracker

Tracking algorithm:
1.) Localize the target
2.) Update visual model



Implement tracker a separate class

- Inherits basic `Tracker` class (in ex2_utils.py)
- See how NCC tracker is implemented (ncc_tracker_example.py)

Mean-shift: tracker

Frame 1: initialization

Extract histogram q
using kernel
(typically: Epanechnikov)

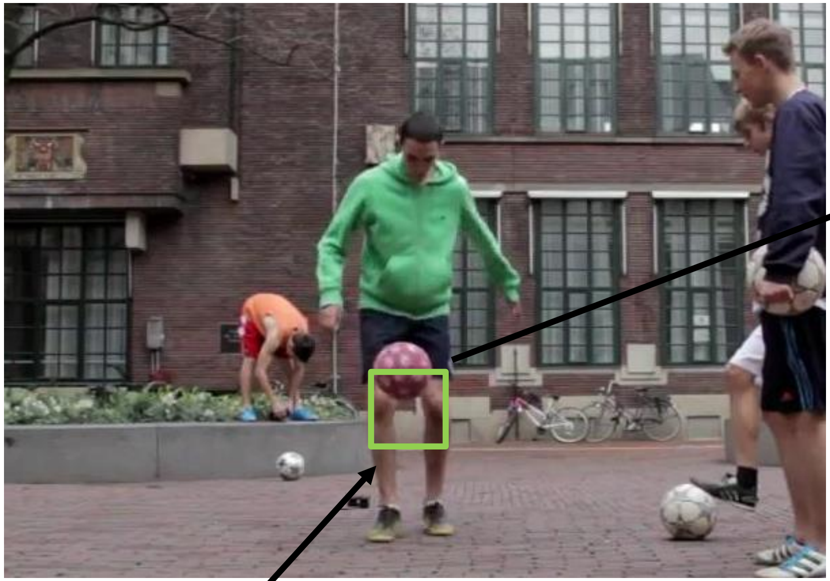
Note: this kernel is NOT the same as
Mean-shift kernel

Histogram extraction kernel and Mean-shift kernel are dependent

Example: if histogram is extracted with epanechnikov kernel,
Mean-shift should use uniform kernel
(see previous slide: kernel-shadow pair)

Mean-shift: tracker

Frame t+1: target localization

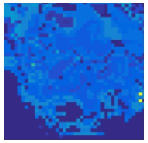


Target position in previous frame

Apply Mean-shift iterations:

Extract histogram p using kernel (see previous slide)

Calculate weights: $\mathrm{v} = \sqrt{\frac{q}{p+eps}}$



w_i = Backproject within extracted patch using weights v

New position using Mean-shift equation

$$x^{(k+1)} = \frac{\sum_{i=1}^{N} x_i w_i g\left(\left\|\frac{x^{(k)} - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{N} w_i g\left(\left\|\frac{x^{(k)} - x_i}{h}\right\|^2\right)}$$

Mean-shift: tracker

Frame t+1: model update

Extract histogram $\tilde{q}$ using kernel

Update model:

$$q_{NEW} = (1-\alpha)q_{OLD} + \alpha\tilde{q}$$

- Try different alphas and report your observations.
- Is it necessary to always update the model?
  - Try setting alpha to 0 and see if performance improves somewhere

Mean-shift: code material

- Some functions have been given to you:
  - `f = generate_responses_1()`
  - `histogram = extract_histogram (image, bins, weights)`
  - `projection = backproject_histogram(image, histogram)`
  - `kernel = create_epanechnik_kernel(w, h, sigma)`
  - `patch, inliers = get_patch(image, center, size)`
- Implement the Mean-shift tracker within Tracker class:
  - Initialize / update tracker
- To test the Mean-shift tracker use the test script

  run_tracker.py

Mean-shift: tips & tricks

- Additional ideas: try different color spaces
  - RGB, HSV, YCbCr, Lab
- Think about efficiency when implementing
  - Do not use for loops
  - Use matrix operations
  - Use meshgrid function
- Implementation details are very important
  - Despite its simplicity, Mean-shift can reach decent performance (with quite low computational cost)