Computer Vision II - Lecture 4

Color based Tracking

29.04.2014

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Course Outline

- Single-Object Tracking
  - Background modeling
  - Template based tracking
  - Color based tracking
  - Contour based tracking
  - Tracking by online classification
  - Tracking-by-detection
- Bayesian Filtering
- Multi-Object Tracking
- Articulated Tracking

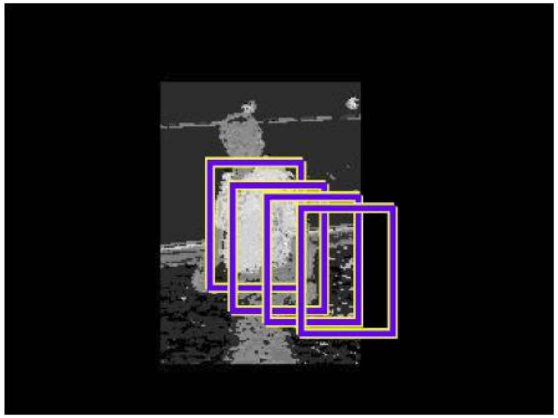
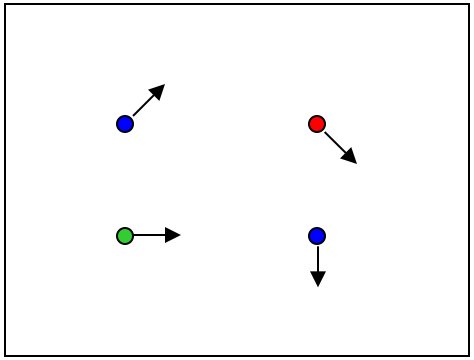
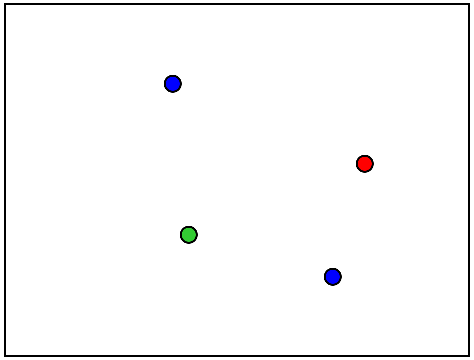


Image source: Robert Collins

Recap: Estimating Optical Flow



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



$I(x,y,t)$

- **Optical Flow**
  - **Given two subsequent frames, estimate the apparent motion field $u(x,y)$ and $v(x,y)$ between them.**
- **Key assumptions**
  - **Brightness constancy: projection of the same point looks the same in every frame.**
  - **Small motion: points do not move very far.**
  - **Spatial coherence: points move like their neighbors.**

Recap: Lucas-Kanade Optical Flow

- **Use all pixels in a $K \times K$ window to get more equations.**
- **Least squares problem:**

$$\begin{bmatrix} I_x(\mathrm{p}_1) & I_y(\mathrm{p}_1) \\ I_x(\mathrm{p}_2) & I_y(\mathrm{p}_2) \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25}) & I_y(\mathrm{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathrm{p}_1) \\ I_t(\mathrm{p}_2) \\ \vdots \\ I_t(\mathrm{p}_{25}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

- **Minimum least squares solution given by solution of**

$$\underset{2\text{x}2}{(A^T A)} \; \underset{2\text{x}1}{d} = \underset{2\text{x}1}{A^T b}$$

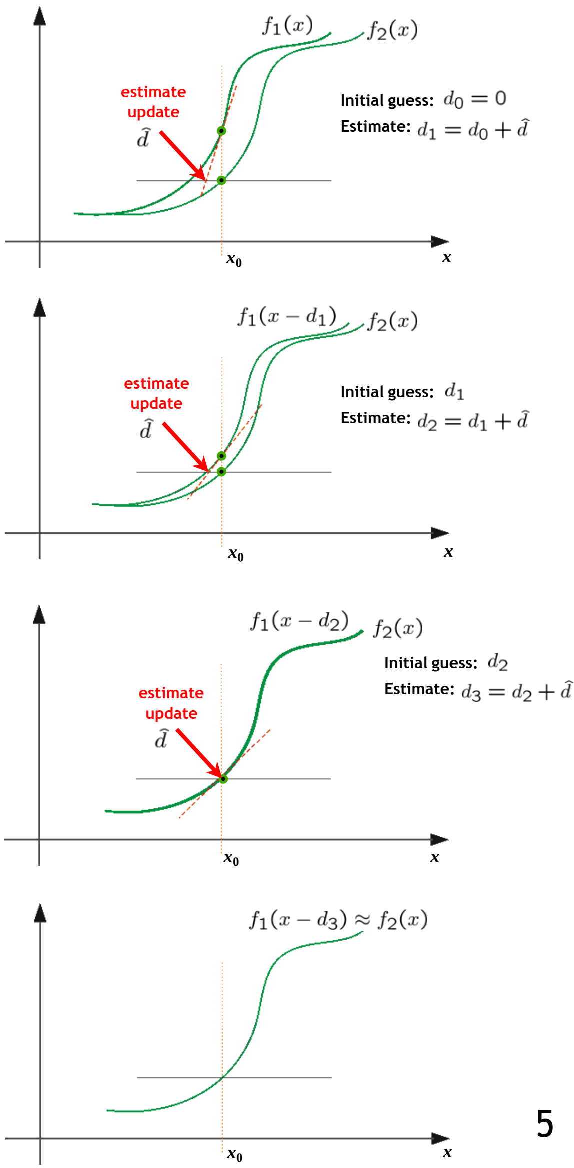
Recall the Harris detector!

$$\underset{A^T A}{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}} \begin{bmatrix} u \\ v \end{bmatrix} = - \underset{A^T b}{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}$$

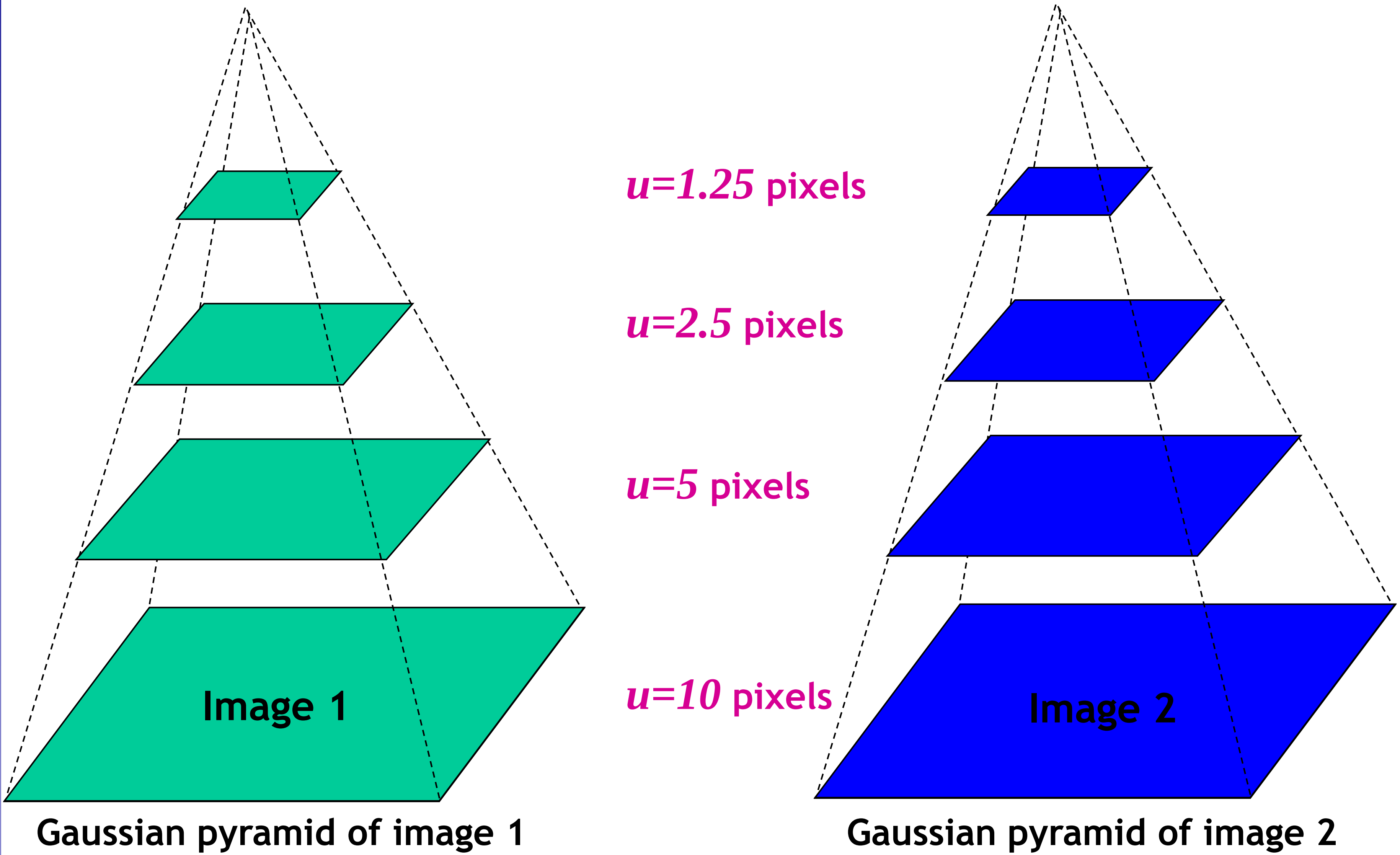
Recap: Iterative Refinement

- Estimate velocity at each pixel using one iteration of LK estimation.
- Warp one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.

- Iterative procedure
  - Results in subpixel accurate localization.
  - Converges for small displacements.

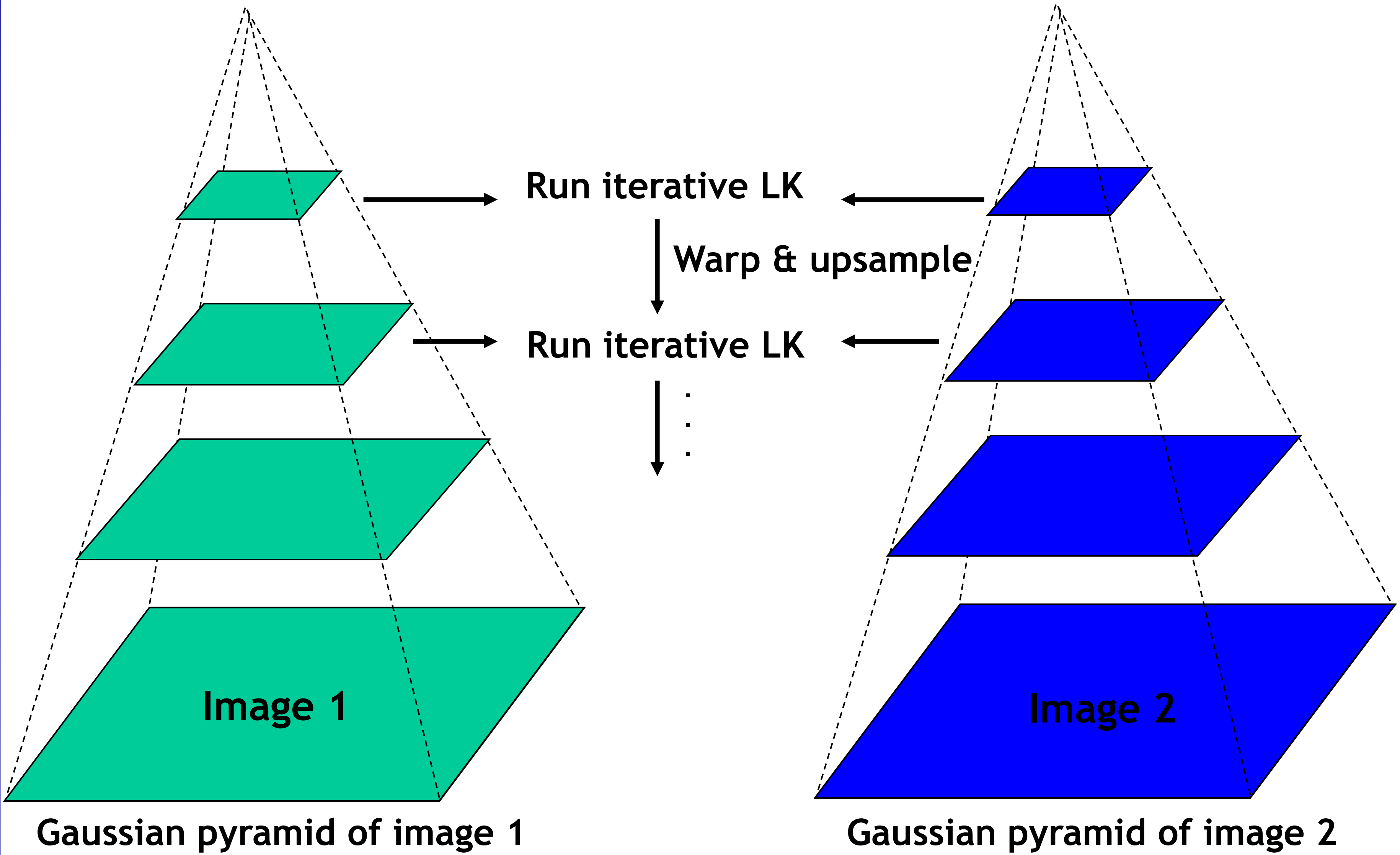


Recap: Coarse-to-fine Optical Flow Estimation



Slide credit: Steve Seitz

Recap: Coarse-to-fine Optical Flow Estimation

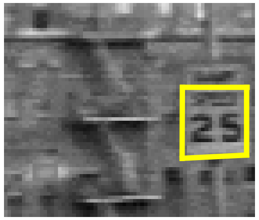


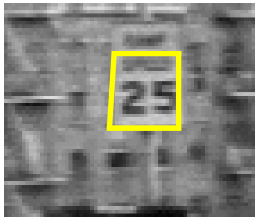
Slide credit: Steve Seitz

Recap: Shi-Tomasi Feature Tracker (→KLT)

- **Idea**
  - Find good features using eigenvalues of second-moment matrix
  - Key idea: "good" features to track are the ones that can be tracked reliably.

- **Frame-to-frame tracking**
  - Track with LK and a pure *translation* motion model.
  - More robust for small displacements, can be estimated from smaller neighborhoods (e.g., 5×5 pixels).





- **Checking consistency of tracks**
  - *Affine* registration to the first observed feature instance.
  - Affine model is more accurate for larger displacements.
  - Comparing to the first frame helps to minimize drift.

J. Shi and C. Tomasi. Good Features to Track. CVPR 1994.

Recap: General LK Image Registration

- **Goal**
  - Find the warping parameters $\mathbf{p}$ that minimize the sum-of-squares intensity difference between the template image $T(\mathbf{x})$ and the warped input image $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$.

- **LK formulation**
  - Formulate this as an optimization problem

$$\arg\min_{\mathbf{p}} \sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x}) \right]^2$$

  - We assume that an initial estimate of $\mathbf{p}$ is known and iteratively solve for increments to the parameters $\Delta\mathbf{p}$:

$$\arg\min_{\Delta\mathbf{p}} \sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) - T(\mathbf{x}) \right]^2$$

Recap: Step-by-Step Derivation

- **Key to the derivation**
  - Taylor expansion around $\Delta\mathbf{p}$

$$I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) \approx I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} + \mathcal{O}(\Delta\mathbf{p}^2)$$

$$= I(\mathbf{W}([x,y];p_1,\ldots,p_n))$$

$$+ \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial W_x}{\partial p_1} & \frac{\partial W_x}{\partial p_2} & \cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \frac{\partial W_y}{\partial p_2} & \cdots & \frac{\partial W_y}{\partial p_n} \end{bmatrix} \begin{bmatrix} \Delta p_1 \\ \Delta p_2 \\ \vdots \\ \Delta p_n \end{bmatrix}$$

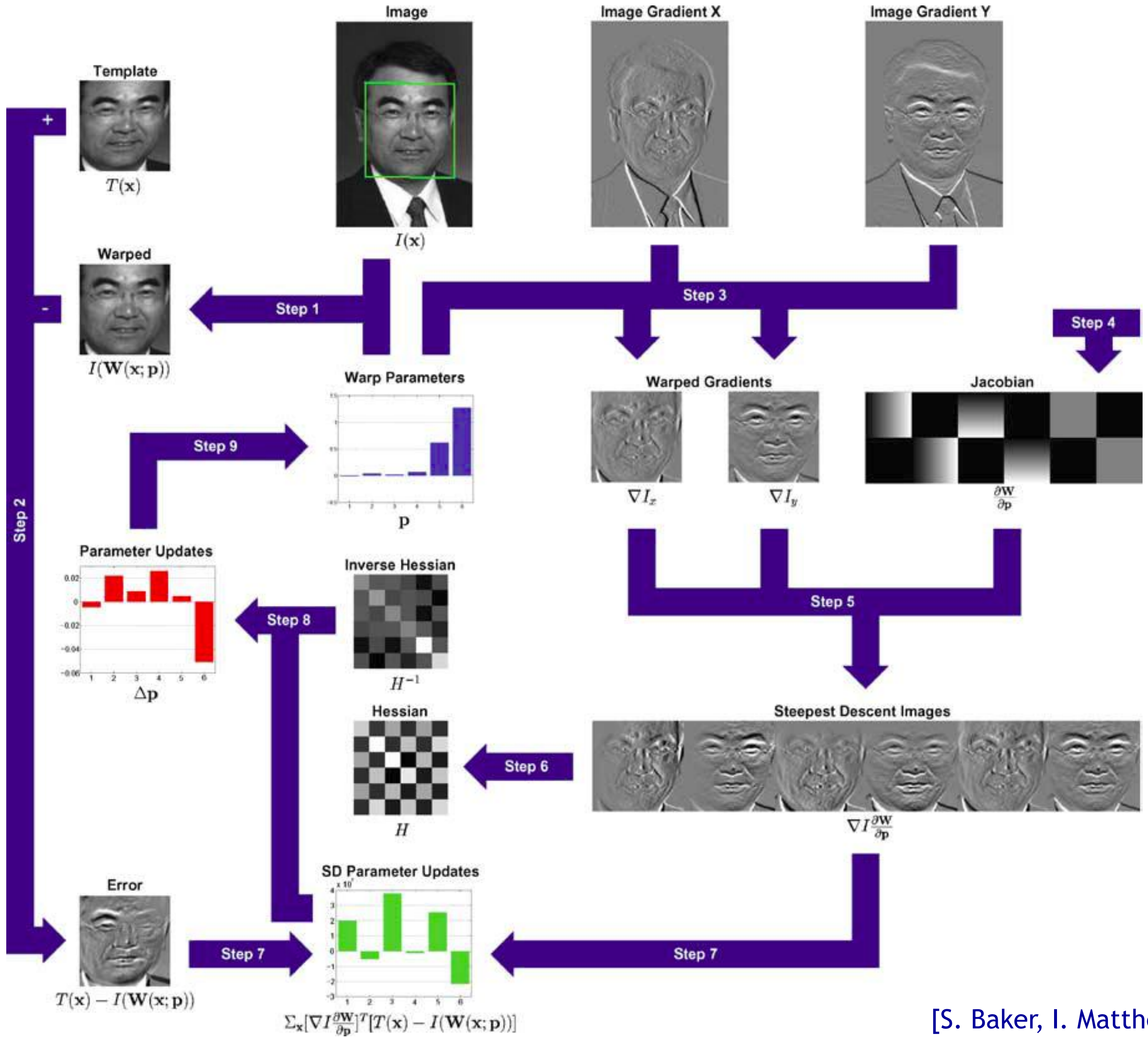
Gradient	Jacobian	Increment parameters to solve for
∇I	$\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$	$\Delta\mathbf{p}$

Recap: LK Algorithm

- Iterate
  - Warp I to obtain $I(\mathbf{W}([x,\, y];\, \mathbf{p}))$
  - Compute the error image $T([x,\, y]) - I(\mathbf{W}([x,\, y];\, \mathbf{p}))$
  - Warp the gradient ∇I with $\mathbf{W}([x,\, y];\, \mathbf{p})$
  - Evaluate $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $([x,\, y];\, \mathbf{p})$ (Jacobian)
  - Compute steepest descent images $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$
  - Compute Hessian matrix $\mathbf{H} = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]$
  - Compute $\sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T \left[T([x,y]) - I(\mathbf{W}([x,y];\mathbf{p}))\right]$
  - Compute $\Delta \mathbf{p} = \mathbf{H}^{-1} \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T \left[T([x,y]) - I(\mathbf{W}([x,y];\mathbf{p}))\right]$
  - Update the parameters $\mathbf{p} \leftarrow \mathbf{p} + \Delta \mathbf{p}$
- Until $\Delta \mathbf{p}$ magnitude is negligible

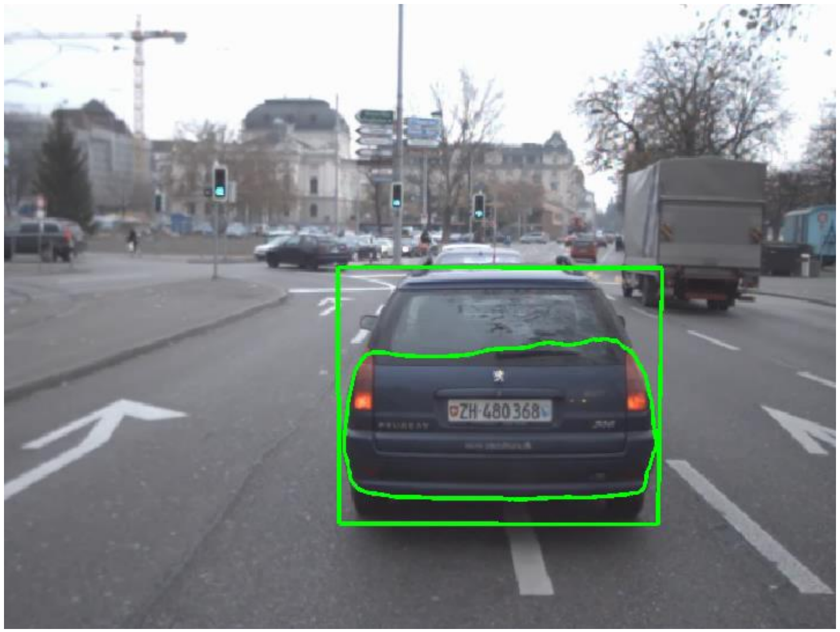
[S. Baker, I. Matthews, IJCV'04]

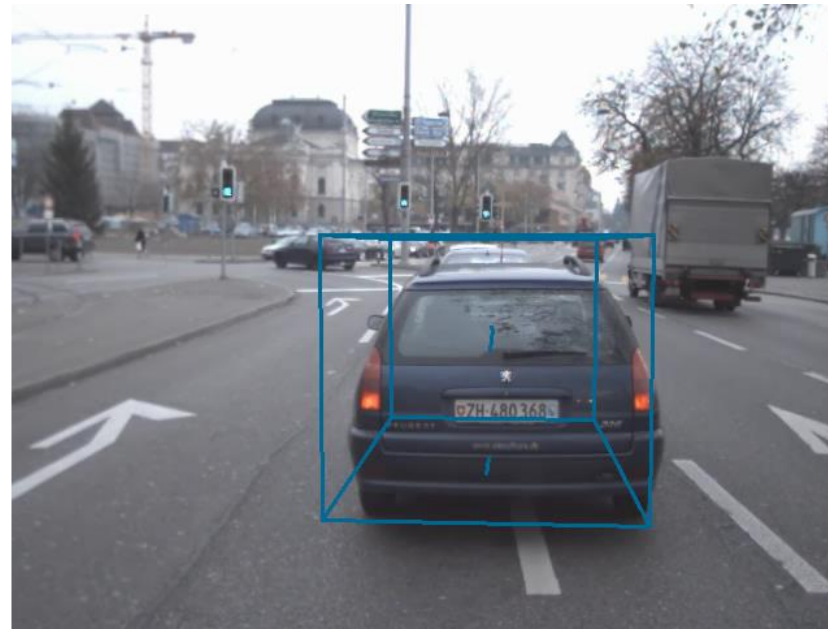
Recap: LK Algorithm Visualization



[S. Baker, I. Matthews, IJCV'04]

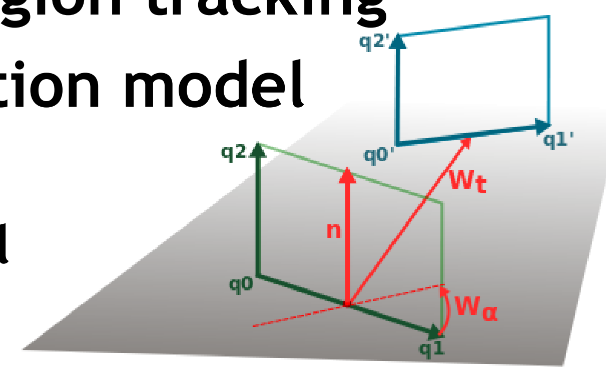
Example of a More Complex Warping Function





- Encode geometric constraints into region tracking
- Constrained homography transformation model
  - Translation parallel to the ground plane
  - Rotation around the ground plane normal
  - $\mathbf{W}(\mathbf{x}) = \mathbf{W}_{obj}\ \mathbf{P}\ \mathbf{W}_t\ \mathbf{W}_\alpha\ \mathbf{Q}\ \mathbf{x}$

$\Rightarrow$ Input for high-level tracker with car steering model.



[E. Horbert, D. Mitzel, B. Leibe, DAGM'10]

Today: Color based Tracking

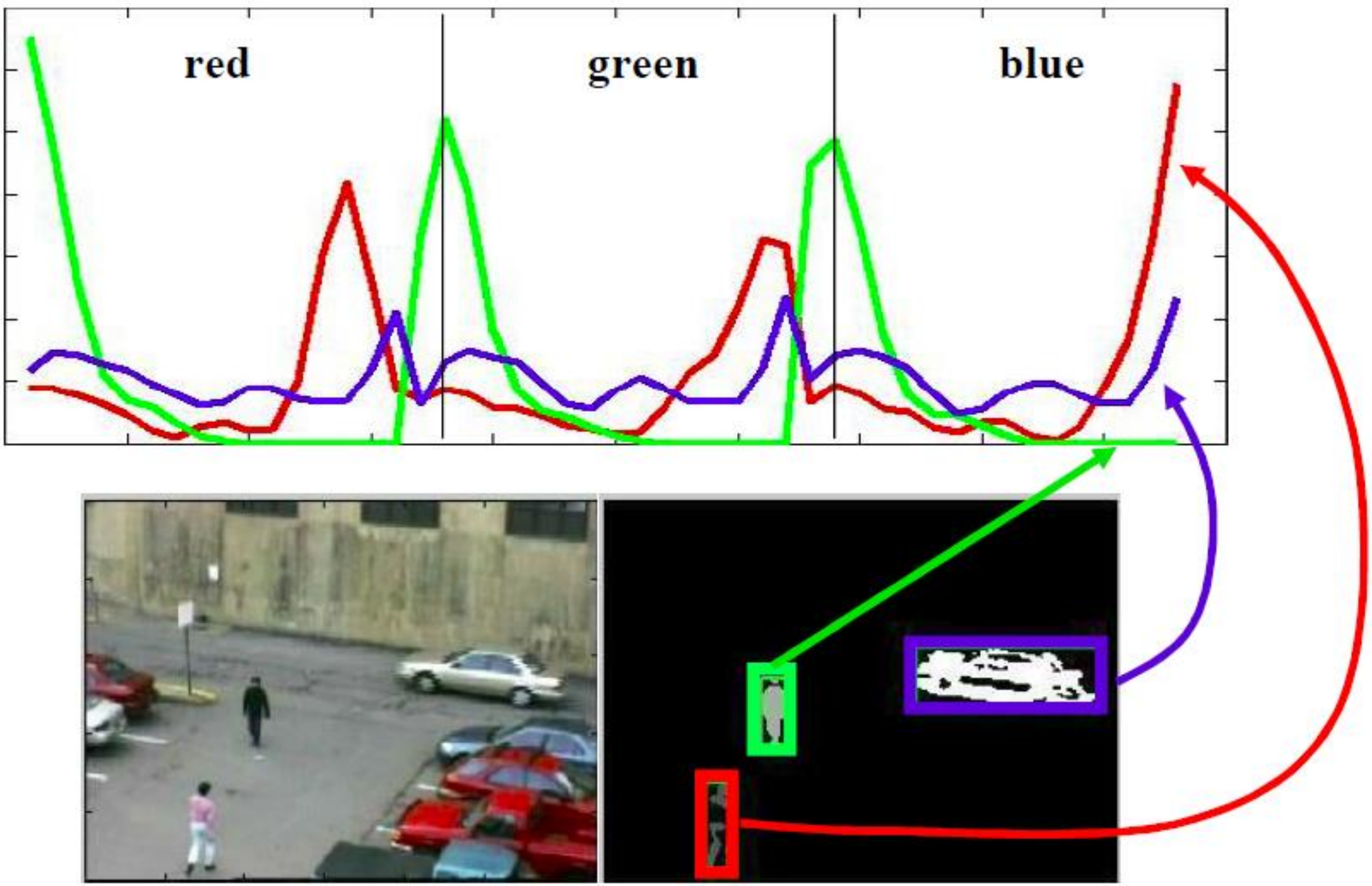


Image source: Robert Collins

Topics of This Lecture

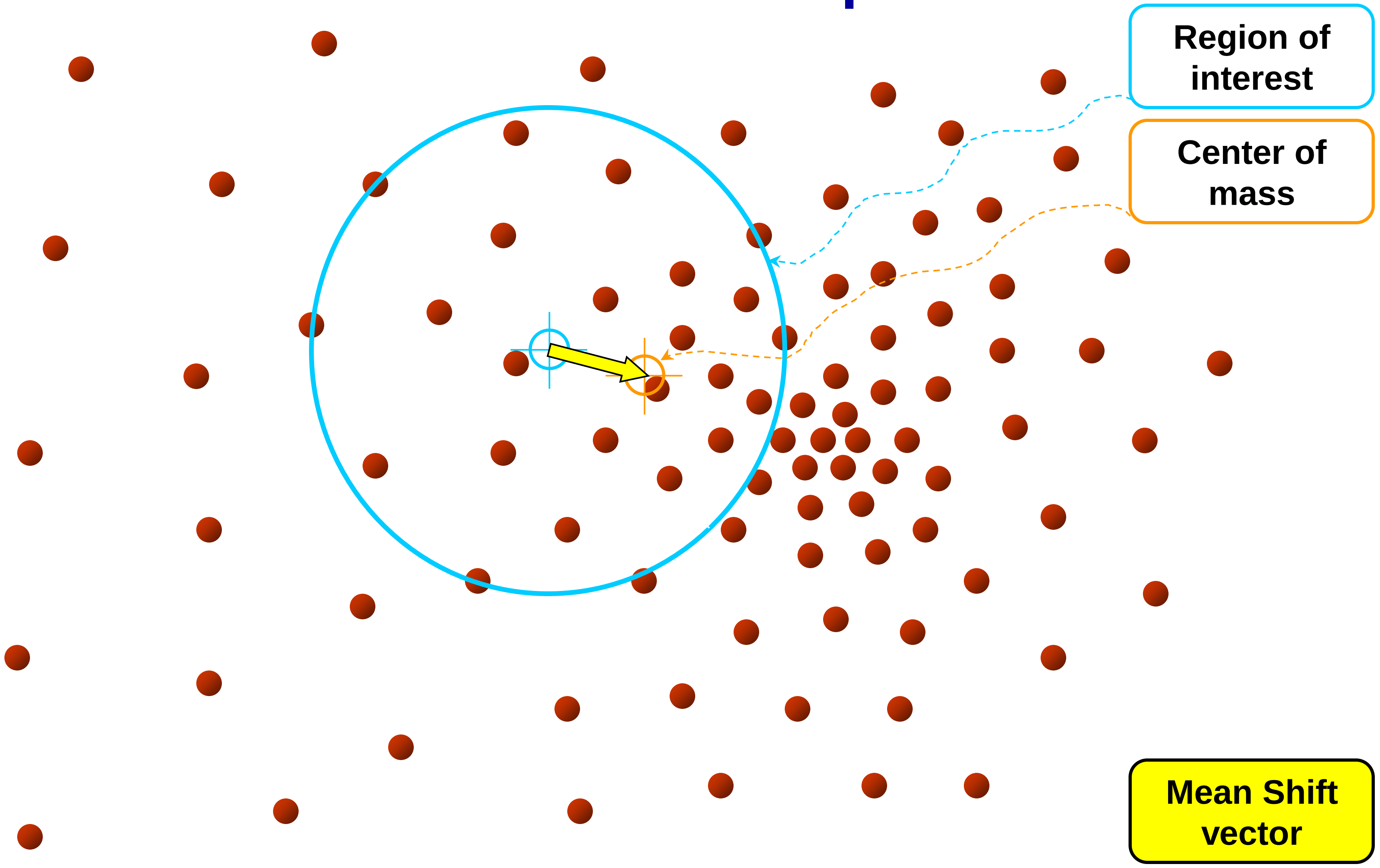
- Mean-Shift
  - Mean-shift mode estimation
  - Using mean-shift on color images
- Mean-Shift with Explicit Weight Images
  - Histogram backprojection
  - CAMshift approach
- Mean-Shift with Implicit Weight Images
  - Comaniciu's approach
  - Bhattacharyya distance
  - Gradient ascent
- Comparison
  - Qualitative intuition

Mean-Shift

- Mean-Shift Tracking
  - Efficient approach to tracking objects whose appearance is defined by color.
  - Actually, the approach is not limited to color. Can also use texture, motion, etc.

- Popular use for object tracking
  - Very simple to implement
  - Non-parametric method, does not make strong assumptions about the shape of the distribution
  - Suitable for non-static distributions (as typical in tracking)
  - Can be combined with dynamic models (Kalman filters, etc.)
  - Good performance in practice

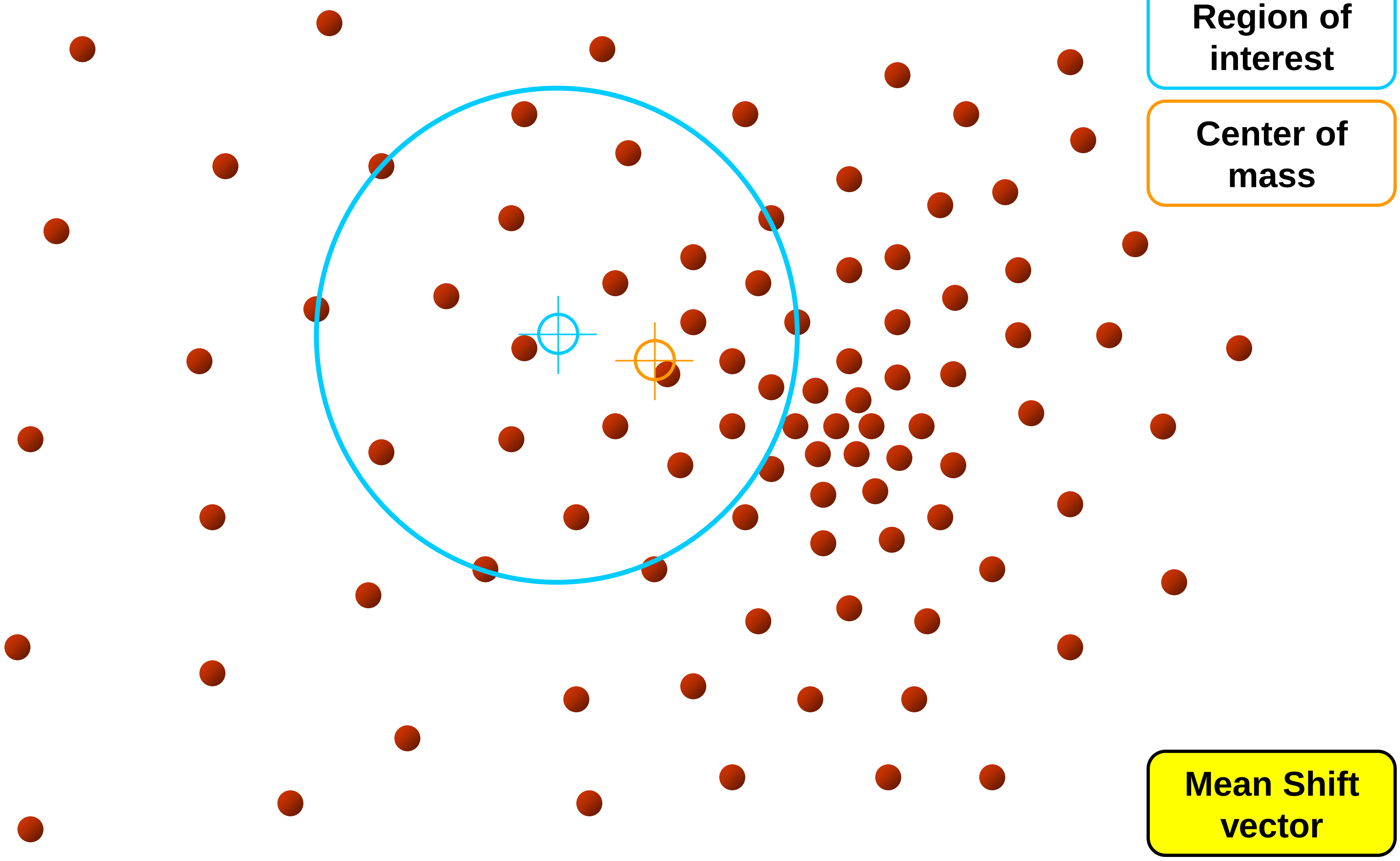
Mean-Shift: Intuitive Description



Objective: Find the densest region

Slide by Y. Ukrainitz & B. Sarel

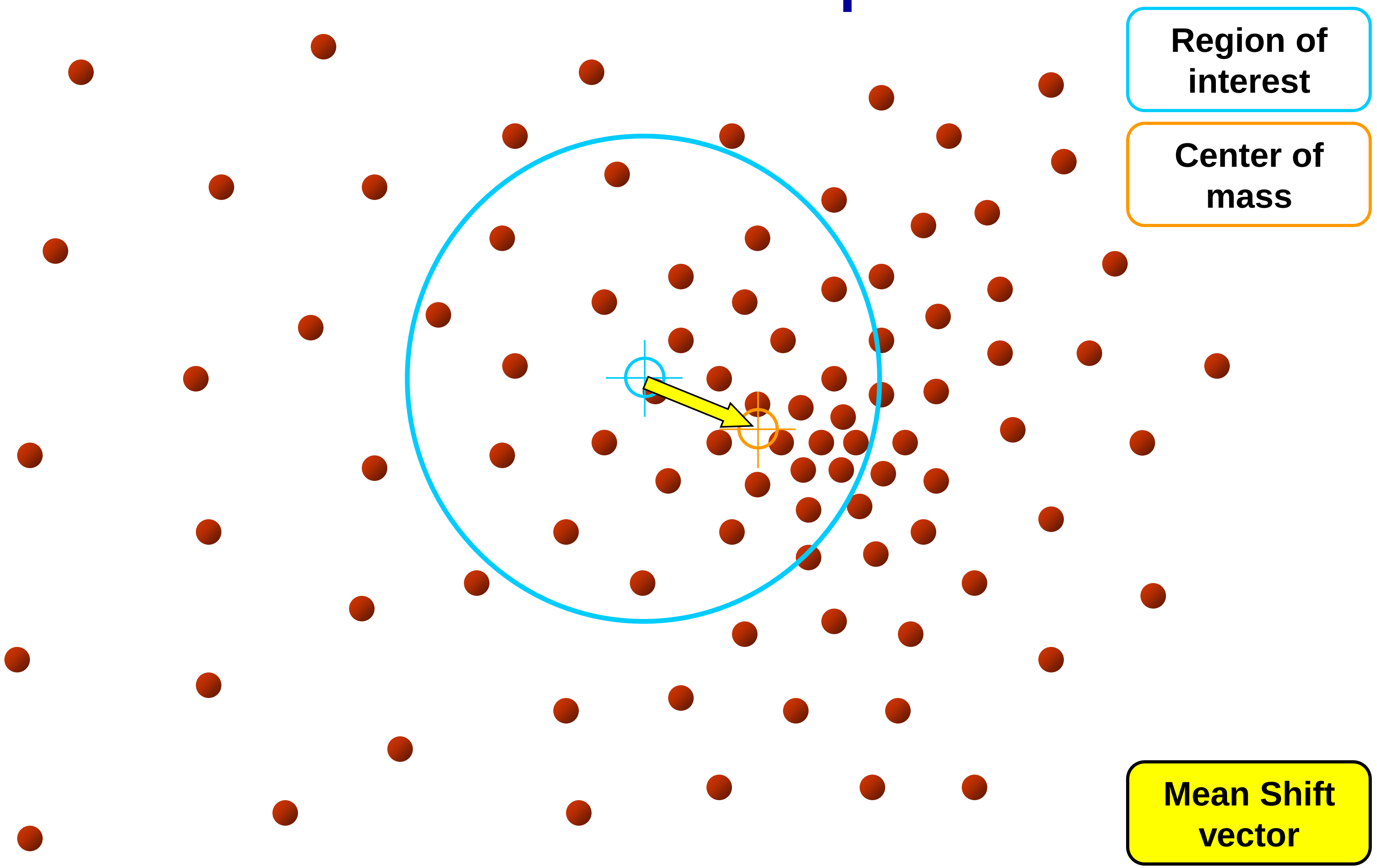
Mean-Shift: Intuitive Description



Objective: Find the densest region

Slide by Y. Ukrainitz & B. Sarel

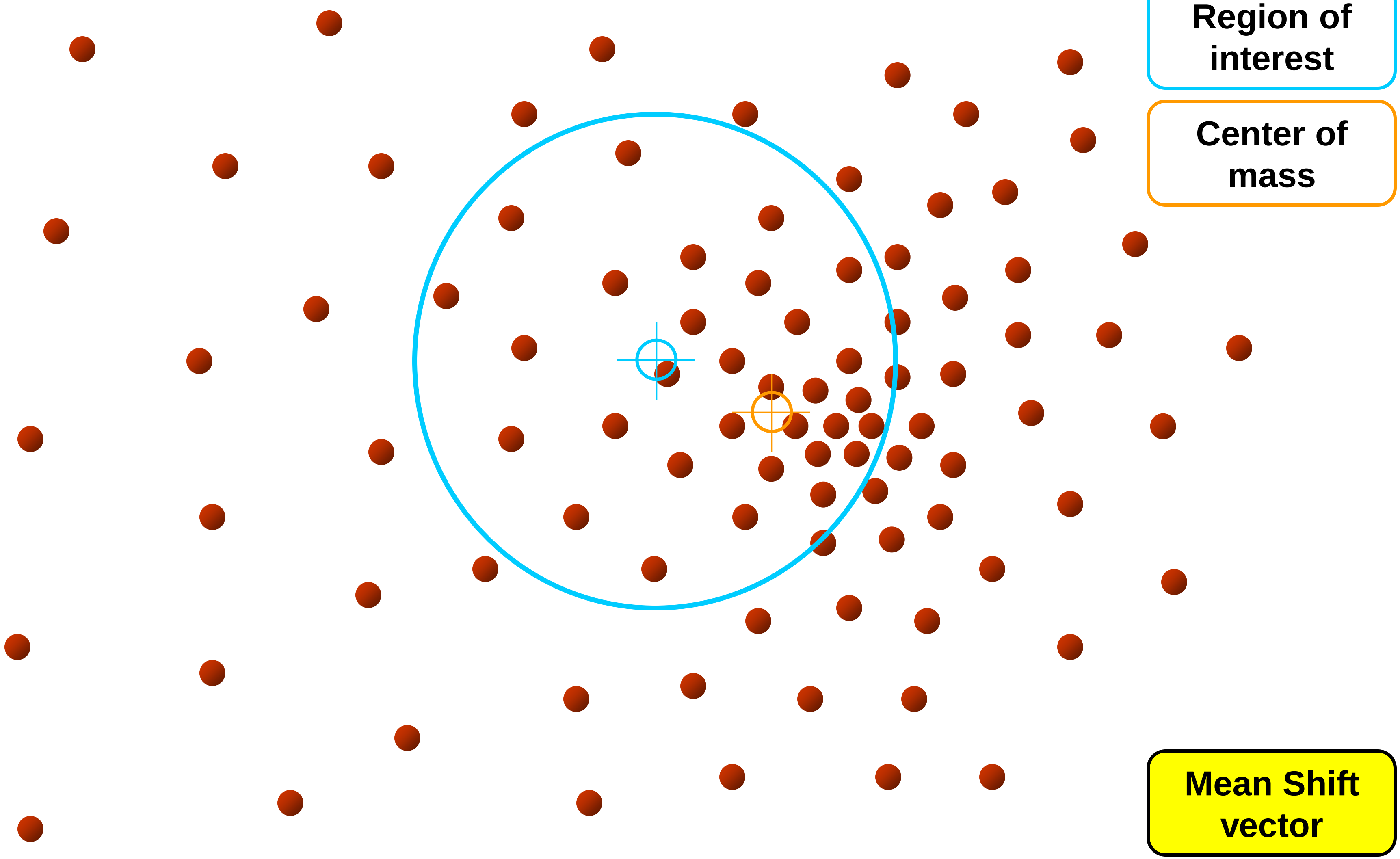
Mean-Shift: Intuitive Description



Objective: Find the densest region

Slide by Y. Ukrainitz & B. Sarel

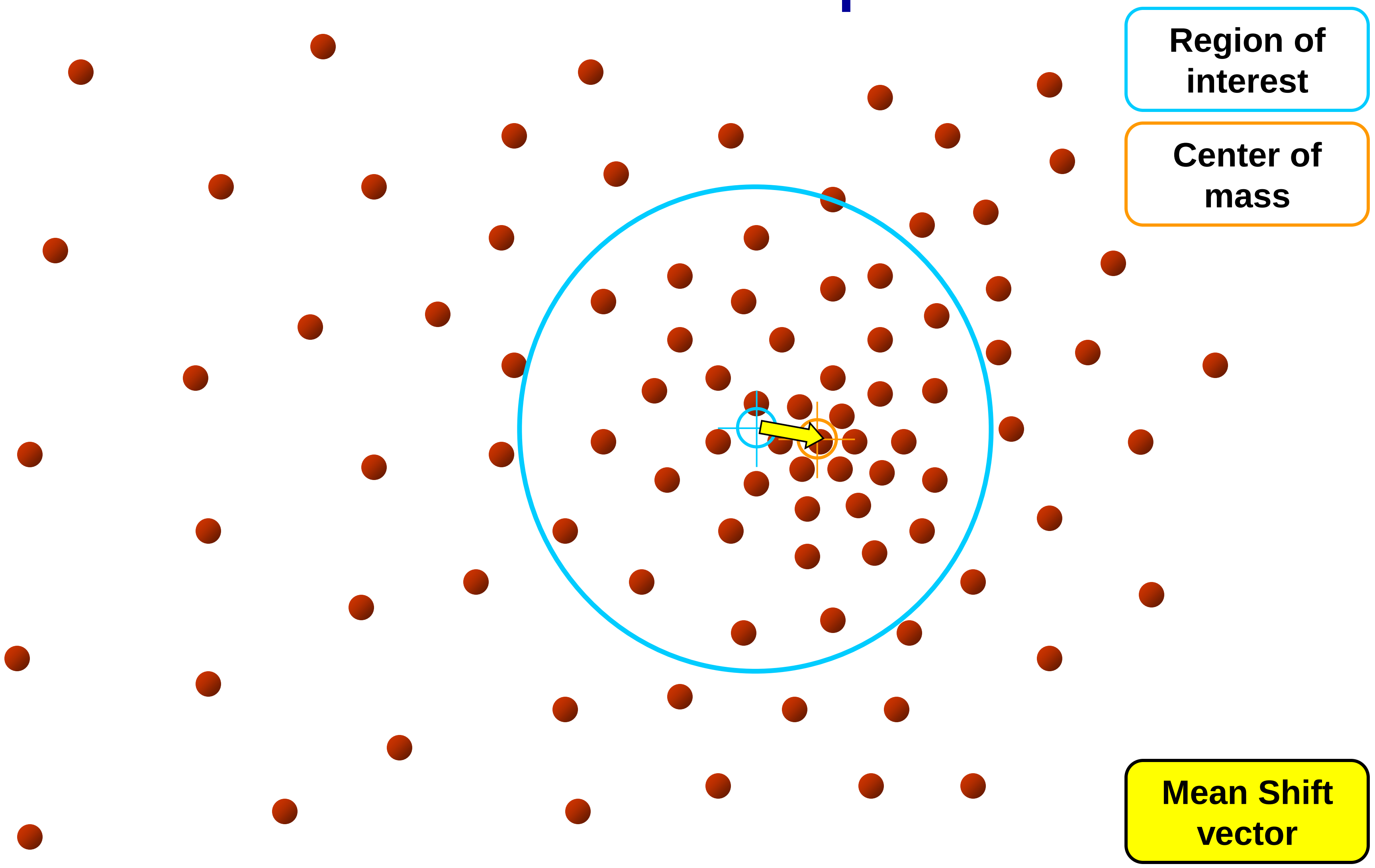
Mean-Shift: Intuitive Description



Objective: Find the densest region

Slide by Y. Ukrainitz & B. Sarel

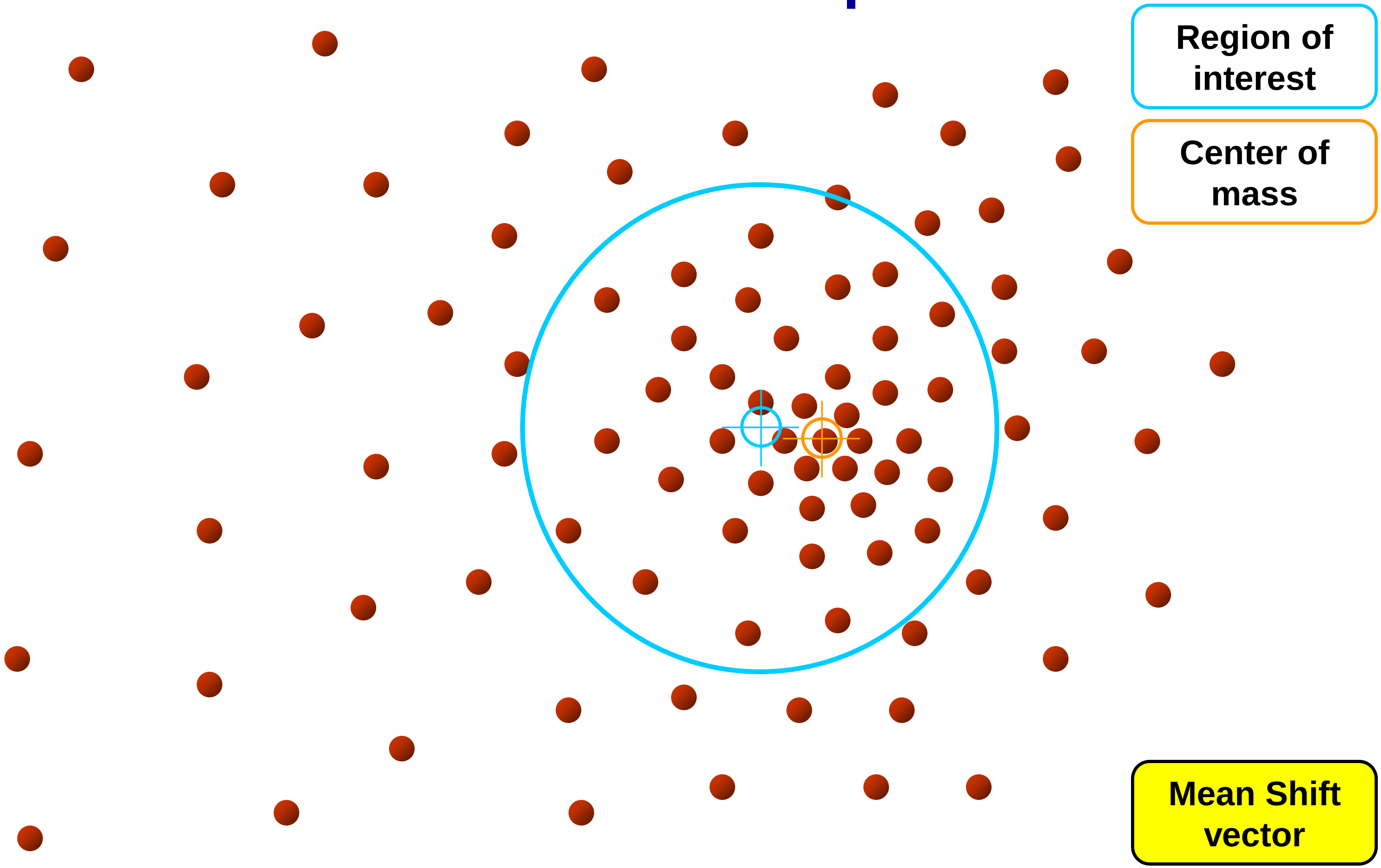
Mean-Shift: Intuitive Description



Objective: Find the densest region

Slide by Y. Ukrainitz & B. Sarel

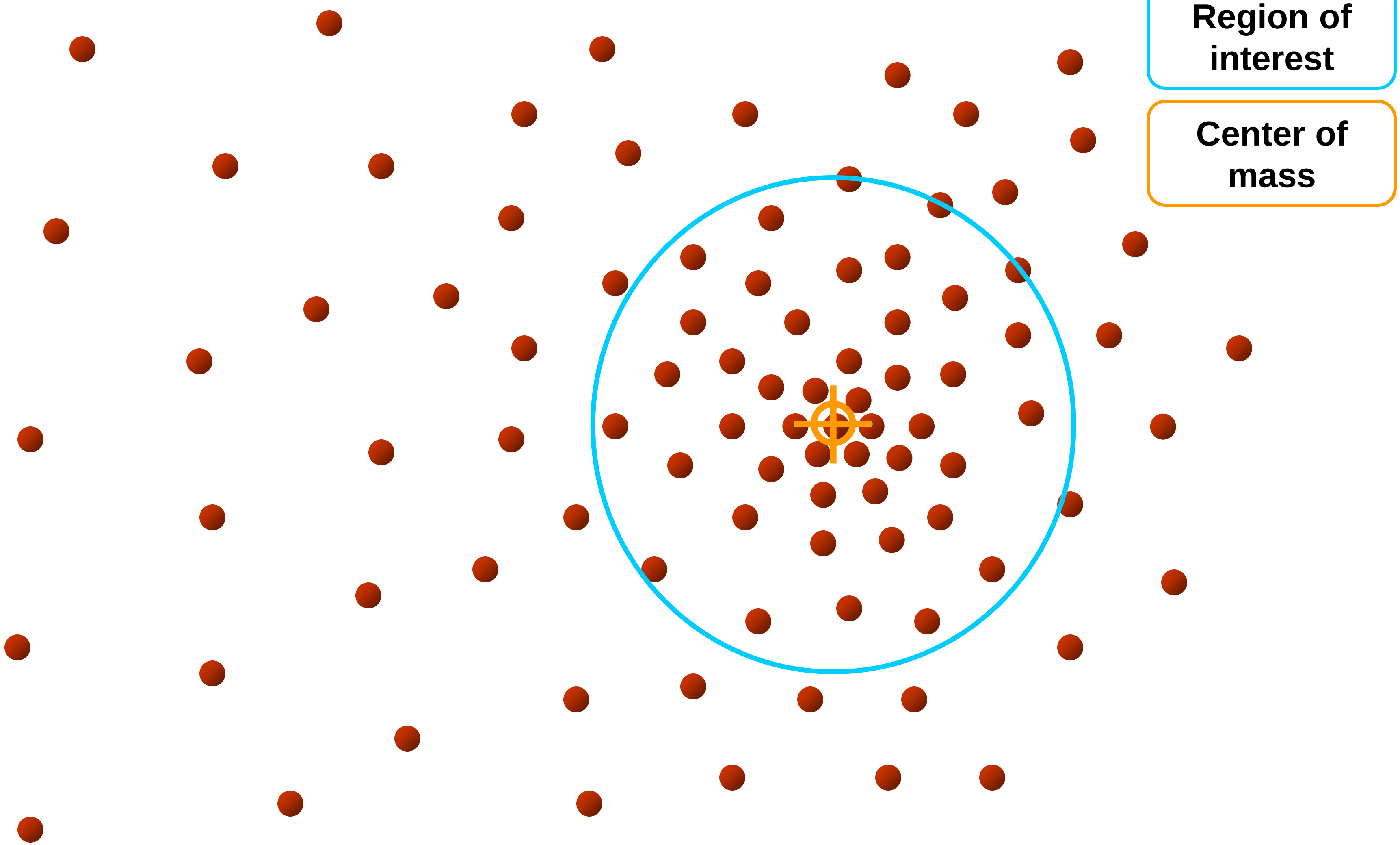
Mean-Shift: Intuitive Description



Objective: Find the densest region

Slide by Y. Ukrainitz & B. Sarel

Mean-Shift: Intuitive Description



Objective: Find the densest region

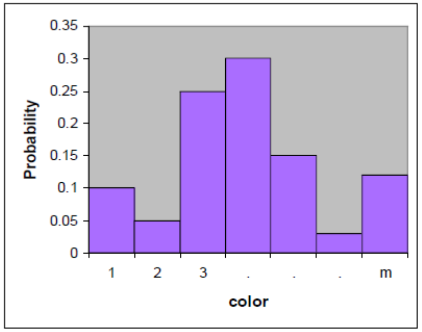
Slide by Y. Ukrainitz & B. Sarel

Using Mean-Shift on Color Models

- Two main approaches

1. Explicit weight images
   - Create a color likelihood image, with pixels weighted by the similarity to the desired color (best for unicolored objects).
   - Use mean-shift to find spatial modes of the likelihood.

2. Implicit weight images
   - Represent color distribution by a histogram.
   - Use mean-shift to find the region that has the most similar color distribution.



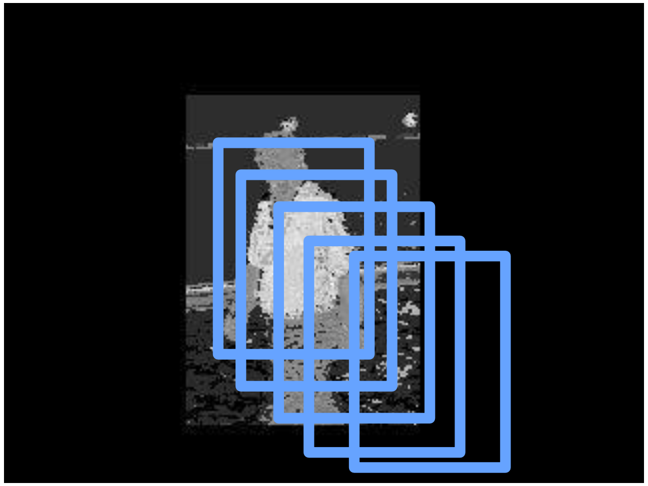
Topics of This Lecture

- Mean-Shift
  - Mean-shift mode estimation
  - Using mean-shift on color images
- **Mean-Shift with Explicit Weight Images**
  - **Histogram backprojection**
  - **CAMshift approach**
- Mean-Shift with Implicit Weight Images
  - Comaniciu’s approach
  - Bhattacharyya distance
  - Gradient ascent
- Comparison
  - Qualitative intuition

Mean-Shift on Weight Images

- Ideal case
  - Want an indicator function that returns 1 for pixels on the tracked object and 0 for all other pixels.
- Instead
  - Compute likelihood maps
  - Value at a pixel is proportional to the likelihood that the pixel comes from the tracked object.
- Likelihood can be based on
  - Color
  - Texture
  - Shape (boundary)
  - Predicted location

Mean-Shift Tracking



- **Idea**
  - Let pixels form a uniform grid of data points.
  - Each pixel has a weight proportional to the likelihood that the pixel is on the object we want to track.
  - Perform standard mean-shift using the weighted set of points.

$$\Delta \mathbf{x} = \frac{\sum_{\mathbf{a}} K(\mathbf{a}-\mathbf{x}) w(\mathbf{a}) (\mathbf{a}-\mathbf{x})}{\sum_{\mathbf{a}} K(\mathbf{a}-\mathbf{x}) w(\mathbf{a})}$$

Mean-Shift Tracking

- A closer look at the procedure...

Kernel weight evaluated at offset $(\mathbf{a}-\mathbf{x})$

Weight from the likelihood image at pixel $\mathbf{a}$

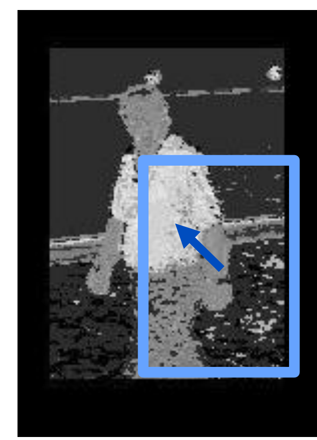
Offset of pixel $\mathbf{a}$ to kernel center $\mathbf{x}$

$$\Delta\mathbf{x} = \frac{\sum_{\mathbf{a}} K(\mathbf{a}-\mathbf{x})w(\mathbf{a})(\mathbf{a}-\mathbf{x})}{\sum_{\mathbf{a}} K(\mathbf{a}-\mathbf{x})w(\mathbf{a})}$$

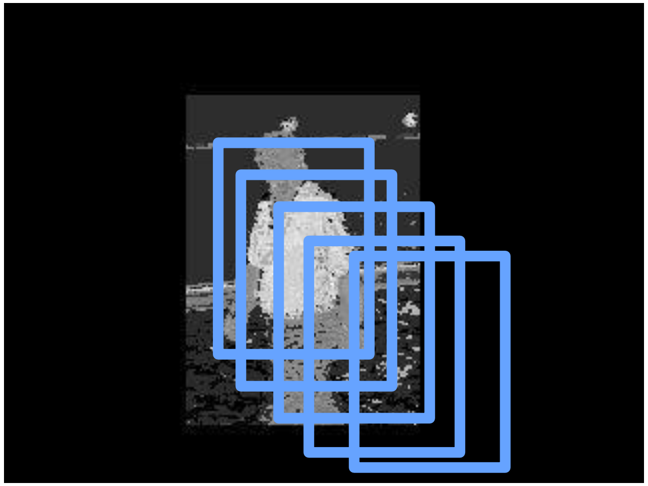
Sum over all pixels $\mathbf{a}$ under kernel K

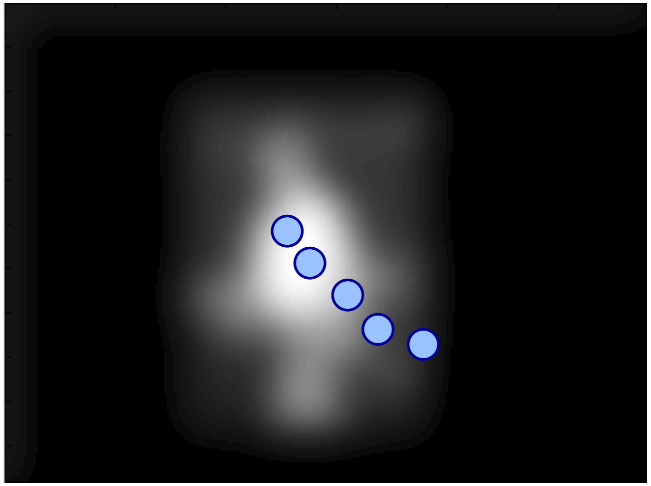
Normalization term

$\Rightarrow$ *Mean-shift computes the weighted mean of all shifts (offsets), weighted by the likelihood under the kernel function.*



Duality Property





- **Duality**
  - **Running mean-shift with kernel K on weight image w is equivalent to performing gradient ascent in a (virtual) image formed by convolving w by some shadow kernel H.**
  - **Note: mode we are looking for is mode of location (x,y) likelihood, NOT mode of color distribution.**

Example: Face Tracking using Mean-Shift

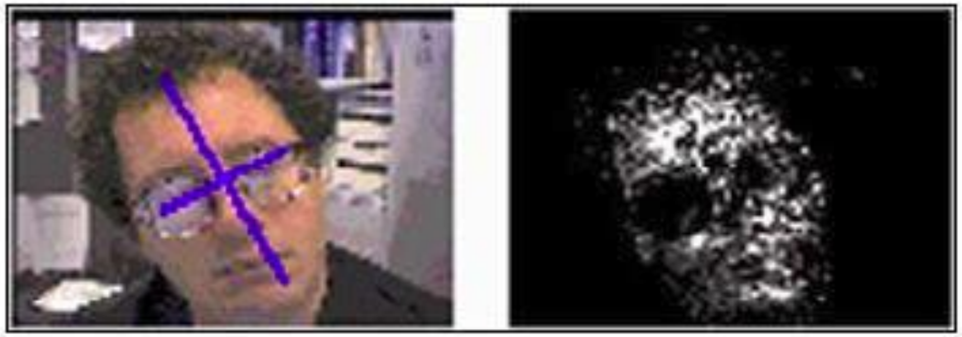


Figure 7: Orientation of the flesh probability distribution marked on the source video image

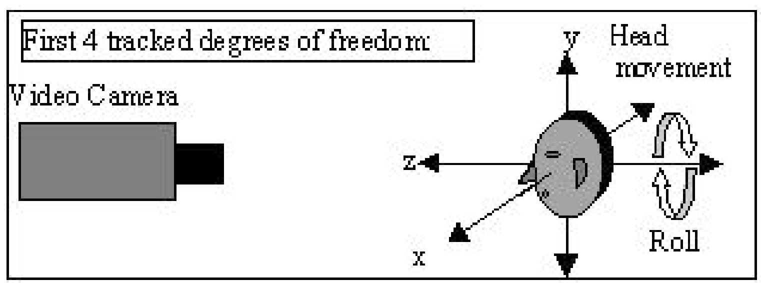


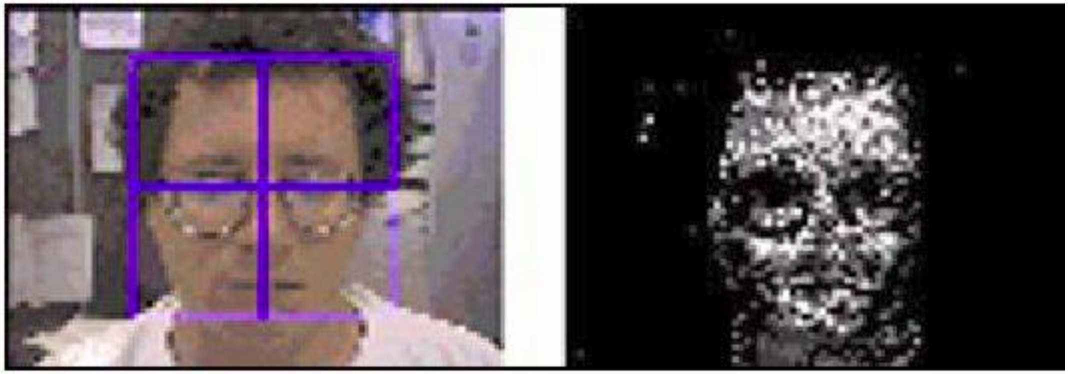
Figure 8: First four head tracked degrees of freedom: X, Y, Z location, and head roll

G. Bradski, Computer Vision Face Tracking for use in a Perceptual User Interface, *IEEE Workshop On Applications of Computer Vision*, Princeton, NJ, 1998, pp.214-219.

Slide credit: Robert Collins

Image source: Gary Bradski

Explicit Weight Images

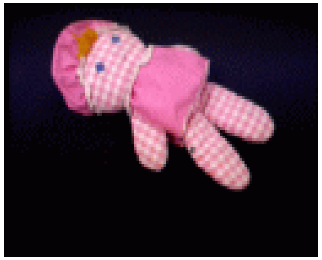


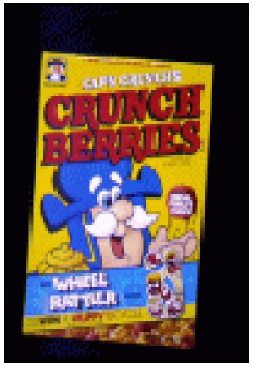
- Histogram backprojection
  - Histogram is an empirical estimate of $p(color \mid object) = p(c \mid o)$
  - Bayes' rule says: $p(o|c) = \dfrac{p(c|o)p(o)}{p(c)}$
  - Simplistic approximation: assume $p(o)/p(c)$ is constant.

  $\Rightarrow$ Use histogram h as a lookup table to set pixel values in the weight image.
  - If pixel maps to histogram bucket i, set weight for pixel to $h(i)$.

Side Note: Color Histograms for Recognition

- Using color histograms for recognition
  - Works surprisingly well
  - In the first paper (1991), 66 objects could be recognized almost without errors



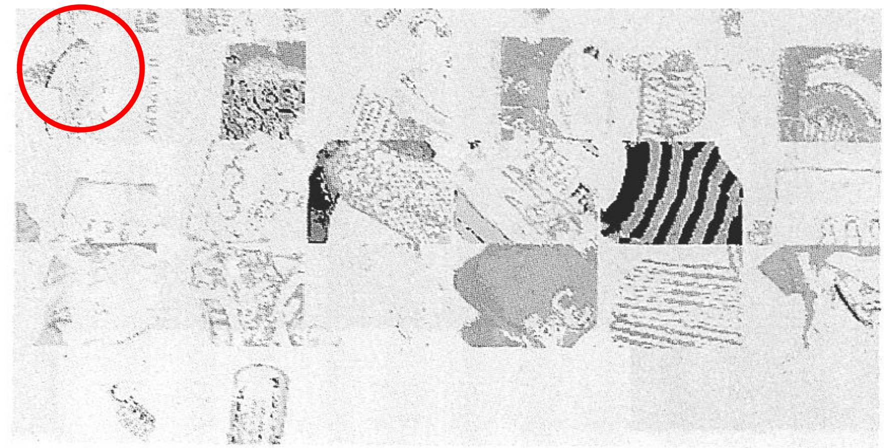
[Swain & Ballard, 1991]

Localization by Histogram Backprojection

- „Where in the image are the colors we‘re looking for?“
  - Query: object with histogram M
  - Given: image with histogram I
- Compute the „ratio histogram“: $$R_i = \min\left(\frac{M_i}{I_i}, 1\right)$$
  - R reveals how important an object color is, relative to the current image.
    - Color is frequent on the object $\Rightarrow$ large M_i
    - Color is frequent in the image $\Rightarrow$ large I_i
  - This value is projected back into the image (i.e. the image values are replaced by the values of R that they index).
  - The result image is convolved with a circular mask the size of the target object.
  - Peaks in the convolved image indicate detected objects.

Does this sound familiar?

Object Localization Results



- **Example result after backprojection**
  - **Looking for blue pullover…**

[Swain & Ballard, 1991]

Bradski's CAMshift

- **Idea**
  - Find x,y location of mode by mean-shift.
  - Determine z, roll angle θ by fitting an ellipse to the mode found using mean-shift.

Visualization: Bradski's CAMshift in Action

Image source: http://docs.opencv.org/trunk/doc/py_tutorials/py_video/py_meanshift/py_meanshift.html

Problem: Scale Changes

- Window always has the same size
  - When the object size changes, does not fit anymore

⇒ Tracking soon diverges...

Image source: http://docs.opencv.org/trunk/doc/py_tutorials/py_video/py_meanshift/py_meanshift.html

Visualization: Scale Adaptation in CAMshift

Mean shift window initialization

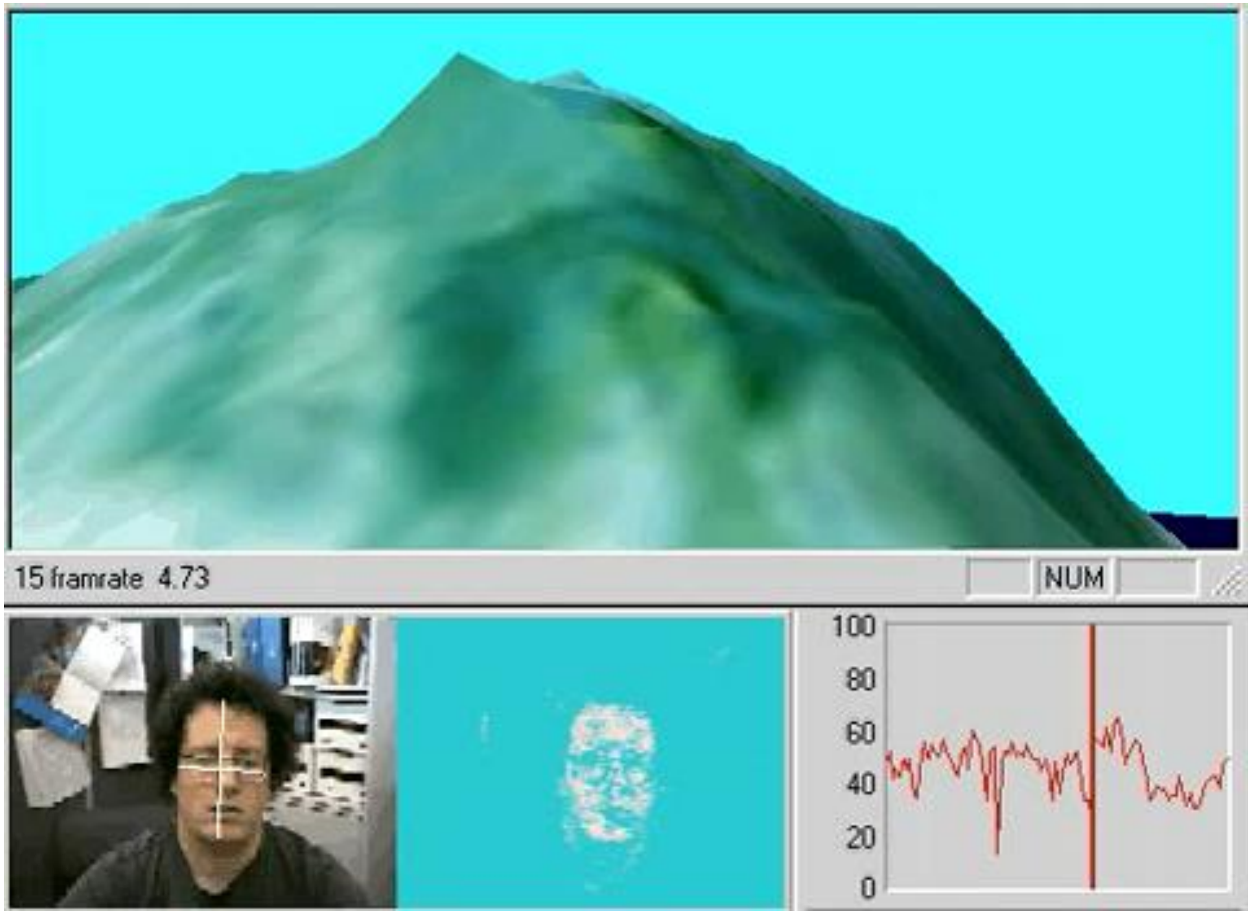
Image source: http://docs.opencv.org/trunk/doc/py_tutorials/py_video/py_meanshift/py_meanshift.html

CAMShift Results

- **Face tracking**
  - Using a skin color model in HSV color space

Video source: Gary Bradski

Applications: Perceptual User Interfaces



- **Head tracking as input modality**
  - **Controlling a flight simulator by head gestures**

Video source: Gary Bradski

Topics of This Lecture

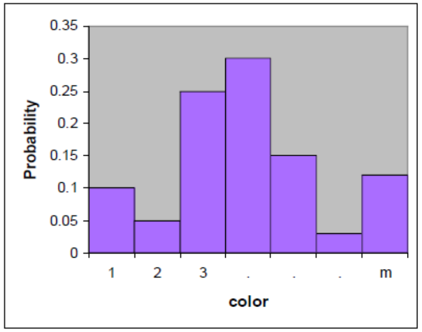
- Mean-Shift
  - Mean-shift mode estimation
  - Using mean-shift on color images
- Mean-Shift with Explicit Weight Images
  - Histogram backprojection
  - CAMshift approach
- **Mean-Shift with Implicit Weight Images**
  - Comaniciu's approach
  - Bhattacharyya distance
  - Gradient ascent
- Comparison
  - Qualitative intuition

Using Mean-Shift on Color Models

- Two main approaches

1. Explicit weight images
   - Create a color likelihood image, with pixels weighted by the similarity to the desired color (best for unicolored objects).
   - Use mean-shift to find spatial modes of the likelihood.

2. Implicit weight images
   - Represent color distribution by a histogram.
   - Use mean-shift to find the region that has the most similar color distribution.



Implicit Weight Images

- Sometimes the weight is not explicitly created
  - Example: Mean-shift Tracking by Comaniciu et al.
  - Weight is embedded into the matching procedure
  - Comes out as a side effect of matching two pdfs.

- Interesting consequence
  - Implicit weight image changes between iterations of mean-shift, as compared to iterating to convergence on an explicit weight image!

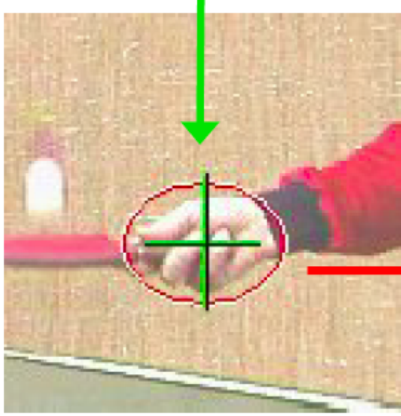
⇒ *We'll take a look at their approach and see how this works.*

D. Comaniciu, V. Ramesh, P. Meer. Kernel-Based Object Tracking, PAMI, Vol. 25(5), pp. 564-575, 2003.

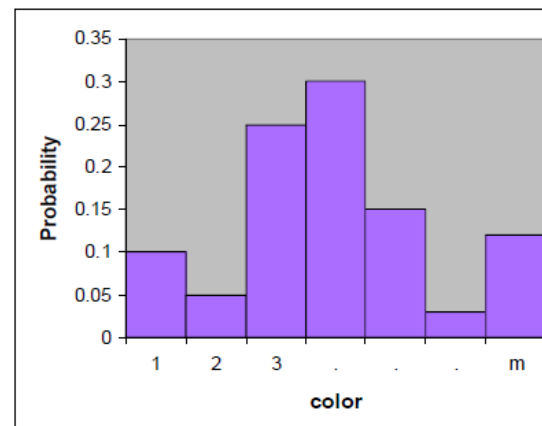
Mean-Shift Object Tracking

- **Main idea: Match the pdf of the target object**

Choose a reference target model → Choose a feature space → Represent the model by its PDF in the feature space

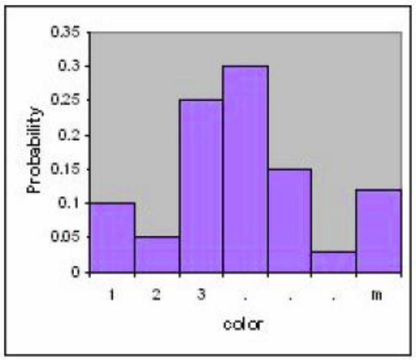


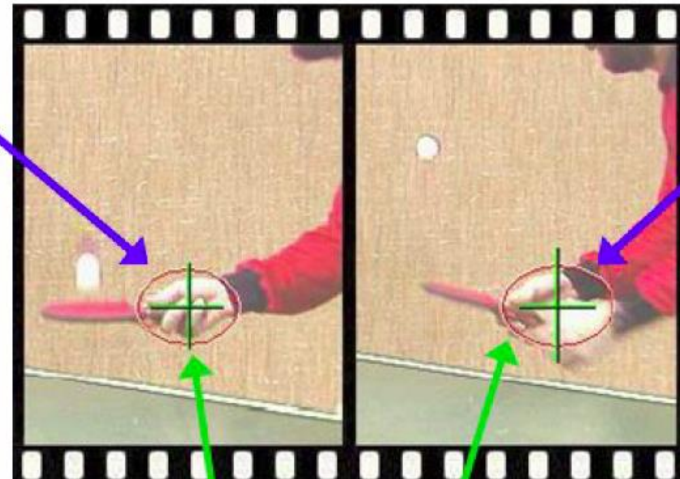
Quantized Color Space



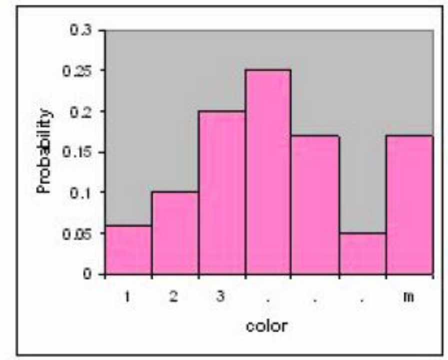
Mean-Shift Object Tracking

Target Model
(centered at 0)





Target Candidate
(centered at y)



$$\vec{q} = \{q_u\}_{u=1..m} \qquad \sum_{u=1}^{m} q_u = 1$$

$$\vec{p}(y) = \{p_u(y)\}_{u=1..m} \qquad \sum_{u=1}^{m} p_u = 1$$

Similarity Function: $f(y) = f\left[\vec{q}, \vec{p}(y)\right]$

Approach

- **Color histogram representation**

target model: $\hat{\mathbf{q}} = \{\hat{q}_u\}_{u=1\ldots m}$ $\sum_{u=1}^{m} \hat{q}_u = 1$

target candidate: $\hat{\mathbf{p}}(\mathbf{y}) = \{\hat{p}_u(\mathbf{y})\}_{u=1\ldots m}$ $\sum_{u=1}^{m} \hat{p}_u = 1\,.$

- **Measuring distances between histograms**
  - Distance as a function of window location $\mathbf{y}$

$$d(\mathbf{y}) = \sqrt{1 - \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right]}\,,$$

  - where $\hat{\rho}(\mathbf{y})$ is the Bhattacharyya coefficient

$$\hat{\rho}(\mathbf{y}) \equiv \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\mathbf{y})\hat{q}_u}\,,$$

Slide credit: Robert Collins

Approach

- Compute histograms via Parzen estimation

$$\hat{q}_u = C \sum_{i=1}^{n} k(\|\mathbf{x}_i^\star\|^2)\delta\left[b(\mathbf{x}_i^\star) - u\right] ,$$

$$\hat{p}_u(\mathbf{y}) = C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h}\right\|^2\right)\delta\left[b(\mathbf{x}_i) - u\right] ,$$

  - where $k(\cdot)$ is some radially symmetric smoothing kernel profile, $\mathbf{x}_i$ is the pixel at location i, and $b(\mathbf{x}_i)$ is the index of its bin in the quantized feature space.

- Consequence of this formulation
  - Gathers a histogram over a neighborhood
  - Also allows interpolation of histograms centered around an off-lattice location.

Finding the Object

- **Goal:**
  - Find the location $\mathbf{y}$ that maximizes the Bhattacharyya coefficient
  - Taylor expansion around current values $p_u(\mathbf{y}_o)$

$$\rho[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}] \approx \frac{1}{2}\sum_{u=1}^{m}\sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u} + \frac{C_h}{2}\sum_{i=1}^{n_h} w_i k\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h}\right\|^2\right)$$

This does not depend on y

⇒ Just need to maximize this. Note: It's a KDE!!!

where $w_i = \sum_{u=1}^{m}\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}}\delta[b(\mathbf{x}_i) - u]$.

Finding the Object

- Taylor expansion around current values $p_u(\mathbf{y}_o)$

$$\rho[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}] \approx \frac{1}{2}\sum_{u=1}^{m}\sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u} + \frac{C_h}{2}\sum_{i=1}^{n_h} w_i k\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h}\right\|^2\right)$$

This does not depend on y

⇒ Just need to maximize this.
Note: It's a KDE!!!

- Find the mode of the second term by mean-shift iterations

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g\left(\left\|\frac{\hat{\mathbf{y}}_0-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\hat{\mathbf{y}}_0-\mathbf{x}_i}{h}\right\|^2\right)} \quad \text{where } g(x) = -k'(x).$$

Slide credit: Robert Collins

Finding the Object

- At each iteration, perform

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g\left(\left\|\frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h}\right\|^2\right)} \quad \text{where } g(x) = -k'(x).$$

  - which is just standard mean-shift on (implicit) weight image w_i.
  - Let's look at the weight image more closely. For each pixel $\mathbf{x}_i$

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}} \boxed{\delta\left[b(\mathbf{x}_i) - u\right]}.$$

This is only 1 once in the summation

⇒ If pixel $\mathbf{x}_i$'s value maps to histogram bucket B, then

$$w_i = \sqrt{(q_B, p_B(\mathbf{y}_0)}$$

Finding the Object

- **Summary**
  - If model histogram is $q_1,\ q_2,\ \ldots,\ q_m$
    and current data histogram is $p_1,\ p_2,\ \ldots,\ p_m$
  - Form weights $q_1/p_1,\ q_2/p_2,\ \ldots,\ q_m/p_m$
  - Do "histogram backprojection" of these values into the image to get the weight image w_i.
    (Note: this is done implicitly)

- **Note**
  - In each iteration, $p_1,\ p_2,\ \ldots,\ p_m$ change, and therefore so does the weight image w_i.

  ⇒ ***Different from applying mean-shift to fixed likelihood image!***

Results: Mean-Shift Tracking

- **Configuration**
  - **Feature space:** $16 \times 16 \times 16$ **quantized RGB**
  - **Target manually selected in 1st frame**
  - **Average mean-shift iterations per frame: 4**

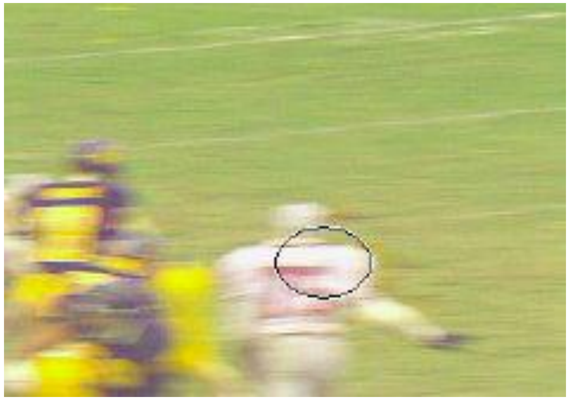
D. Comaniciu, V. Ramesh, P. Meer. Kernel-Based Object Tracking, PAMI, Vol. 25(5), pp. 564-575, 2003.

Results: Mean-Shift Tracking

- **Difficulties**

Partial occlusion

Distraction



Motion blur

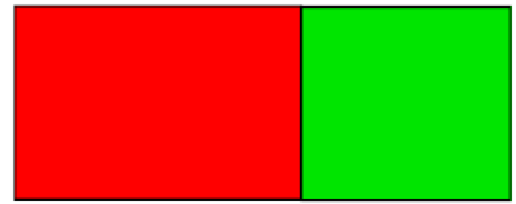
⇒ **Mean-shift still performs robustly despite those.**

Topics of This Lecture

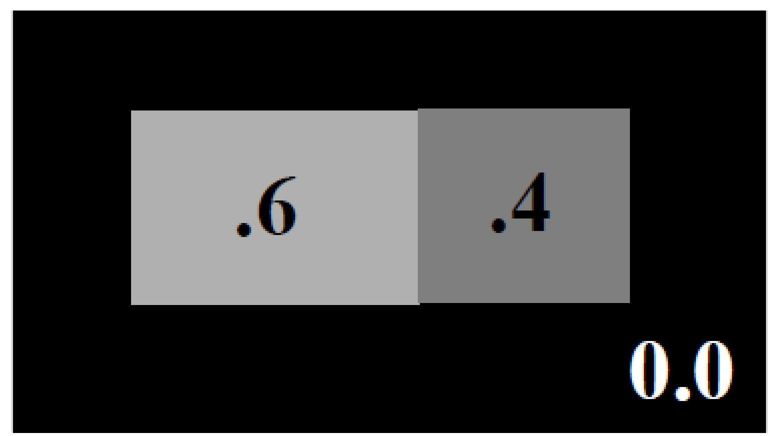
- Mean-Shift
  - Mean-shift mode estimation
  - Using mean-shift on color images
- Mean-Shift with Explicit Weight Images
  - Histogram backprojection
  - CAMshift approach
- Mean-Shift with Implicit Weight Images
  - Comaniciu’s approach
  - Bhattacharyya distance
  - Gradient ascent
- **Comparison**
  - **Qualitative intuition**

Qualitative Intuition

- **Bradski's Mean-Shift procedure**
  - **Assume that an object is 60% red and 40% green.**
  - **I.e., $q_1 = 0.6$, $q_2 = 0.4$, $q_i = 0$ for all other i.**



  - **If we just did histogram backprojection of these likelihood values (a la Bradski), we would get this weight image:**
  - **Mean-shift does a weighted center-of-mass computation at each iteration.**



$\Rightarrow$ ***Window will be biased towards the region of red pixels, since they have higher weight!***

Qualitative Intuition

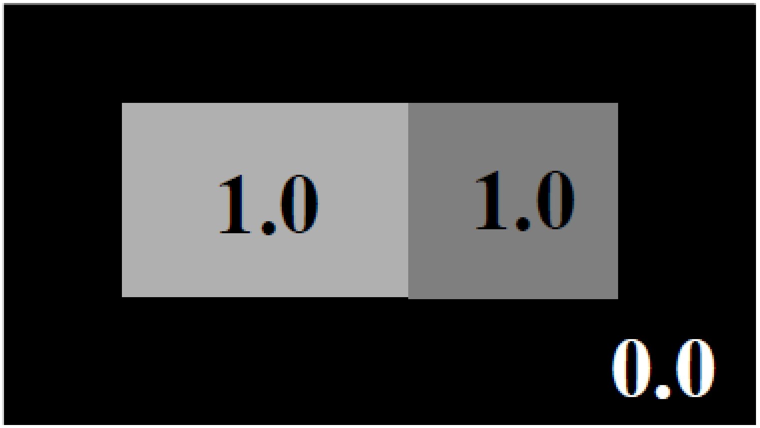
- **Comaniciu's approach**
  - **Let's assume the data histogram is perfectly located**

$\Rightarrow q_1 = 0.6$, $q_2 = 0.4$, $q_i = 0$ **for all other** i**.**

$p_1 = 0.6$, $p_2 = 0.4$, $p_i = 0$ **for all other** i**.**

$\Rightarrow w_1 = \mathrm{sqrt}(0.6/0.6)$, $w_2 = \mathrm{sqrt}(0.4/0.4)$, $w_i = 0$ **for all other** i**.**

$\Rightarrow$ **Resulting weight image:**



$\Rightarrow$ ***Much better!***

$\Rightarrow$ ***Perfect object indicator function.***

References and Further Reading

- The original CAMshift paper
  - G. Bradski, Computer Vision Face Tracking for use in a Perceptual User Interface, IEEE Workshop On Applications of Computer Vision, Princeton, NJ, 1998, pp.214-219.
- The Mean-Shift Tracking paper by Comaniciu
  - D. Comaniciu, V. Ramesh, P. Meer. Kernel-Based Object Tracking, PAMI, Vol. 25(5), pp. 564-575, 2003.