Intelligent Control Systems

Visual Tracking (2)

— Color-based tracking by mean shifting —

Shingo Kagami

Graduate School of Information Sciences,

Tohoku University

swk(at)ic.is.tohoku.ac.jp

http://www.ic.is.tohoku.ac.jp/ja/swk/

Histogram-based Tracking

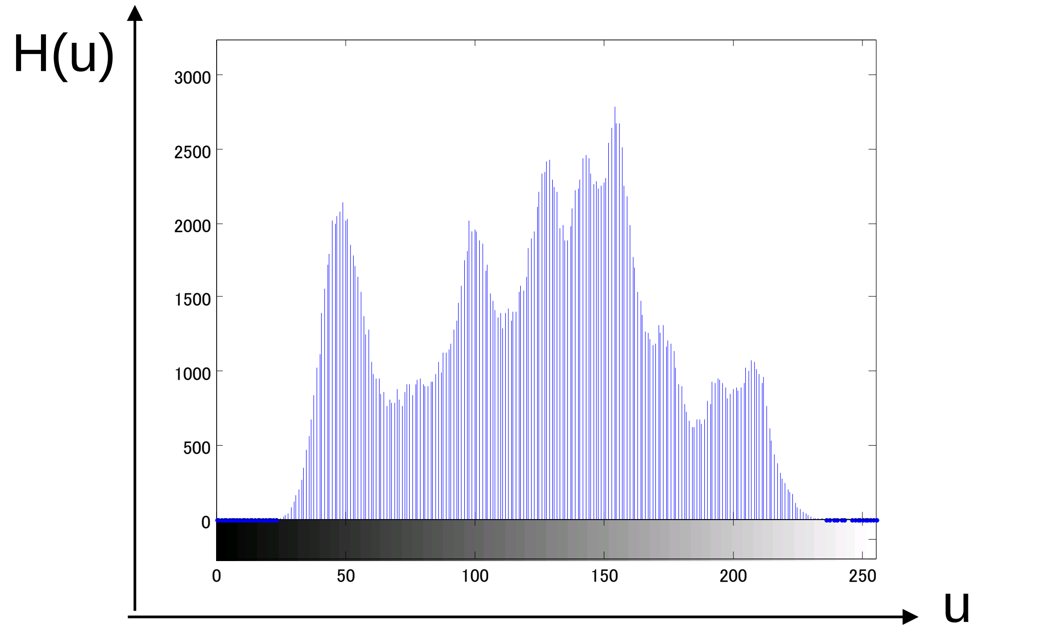
- We have so far seen that tracking consists in:
  evaluation function + optimization method
  - e.g. Lucas-Kanade = SSD + Gauss-Newton

- Today we use similarity of color histograms as evaluation function, which is (almost) invariant with shape deformation
- Mean Shift [Fukunaga 1975] is used as optimization method to minimize an approximation of the color histogram similarity [Comaniciu 2003]

Agenda

- Color Histogram
  - non-weighted and weighted
- Similarity of Histograms
  - Bhattacharyya coefficients
  - approximated for mean shift tracking
- Mean Shift Tracking

(Grayscale) Histogram



$$\boldsymbol{H} = \{H_u\}_{u=1,2,\cdots,m}, \;\; H_u = \sum_{\boldsymbol{x} \in S(u)} 1$$

where S(u) is a set of pixels having values belonging to the bin u

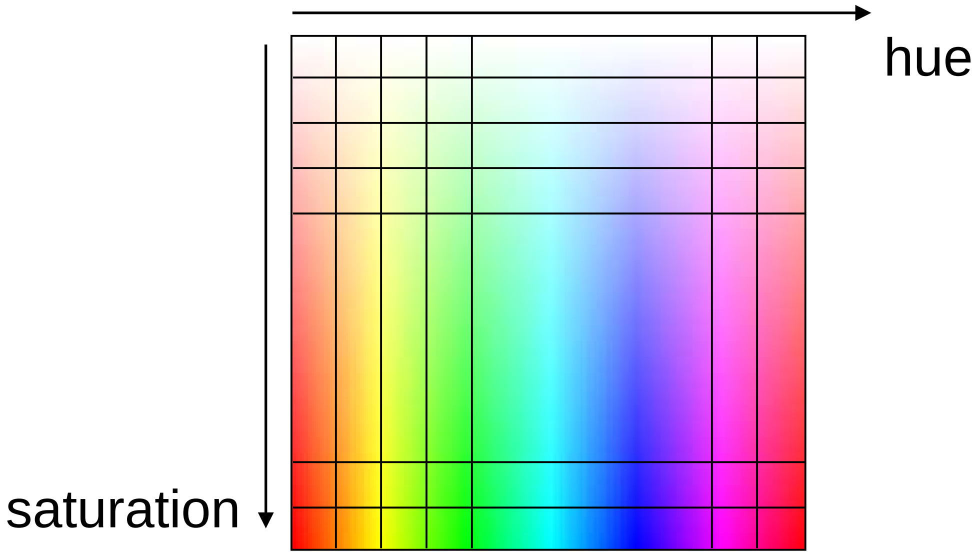
$$\boldsymbol{p} = \{p_u\}, \;\; p_u \propto H_u, \;\; \sum_{u=1}^{m} p_u = 1$$ (normalized histogram)

Color Histograms

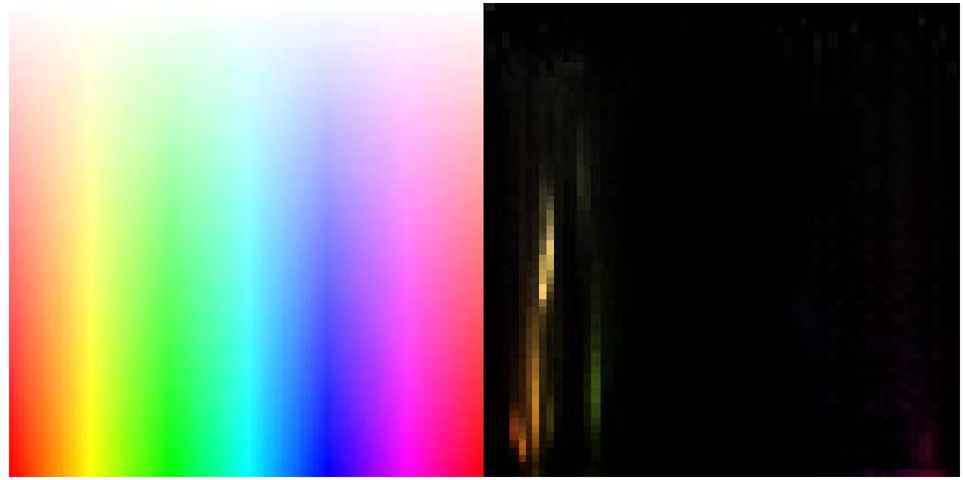
ex1) By splitting each of RGB components into 16 bins, we have histogram over 16 x 16 x 16 bins

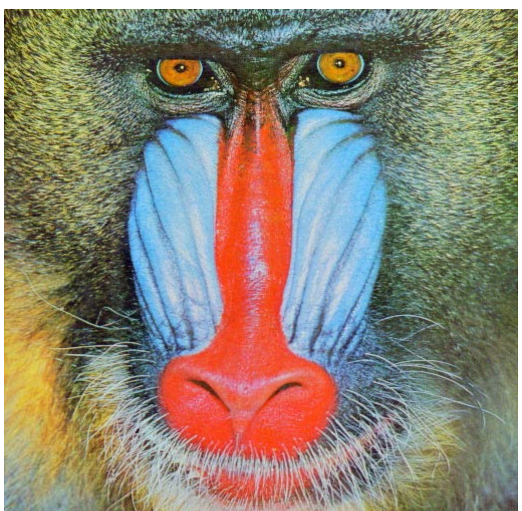
ex2) By splitting each of Hue and Saturation components into 64 bins (and ignoring Value component), we have histogram over 64 x 64 bins

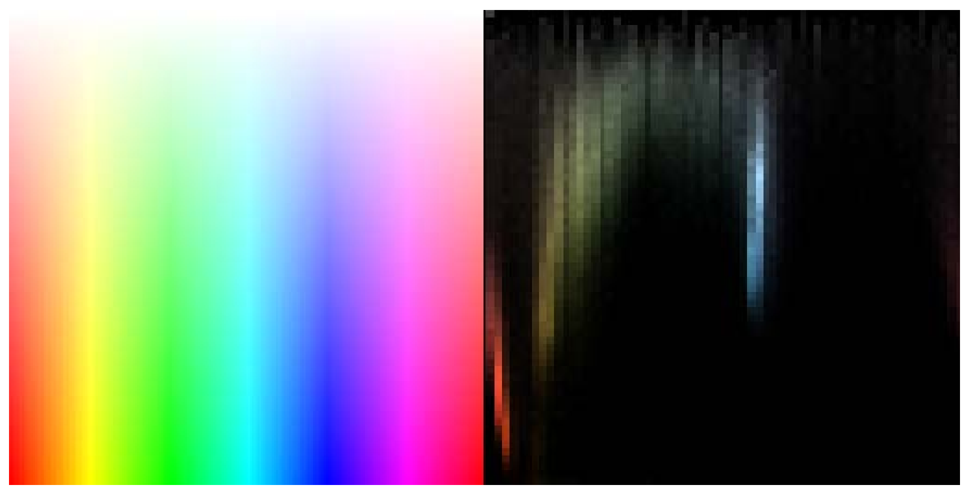
- More unaffected by illumination change



Hue-Saturation Histograms







Similarity of Histograms

- Our objective is to find a region with histogram similar to that of a given model
- How do we measure the similarity?

Bhattacharyya Coefficient

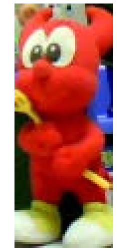
- is a metric for similarity of two probabilistic distributions (and thus, of two normalized histograms) **p** and **q**

$$\rho(\boldsymbol{p}, \boldsymbol{q}) = \sum_{u=1}^{m} \sqrt{p_u q_u}$$

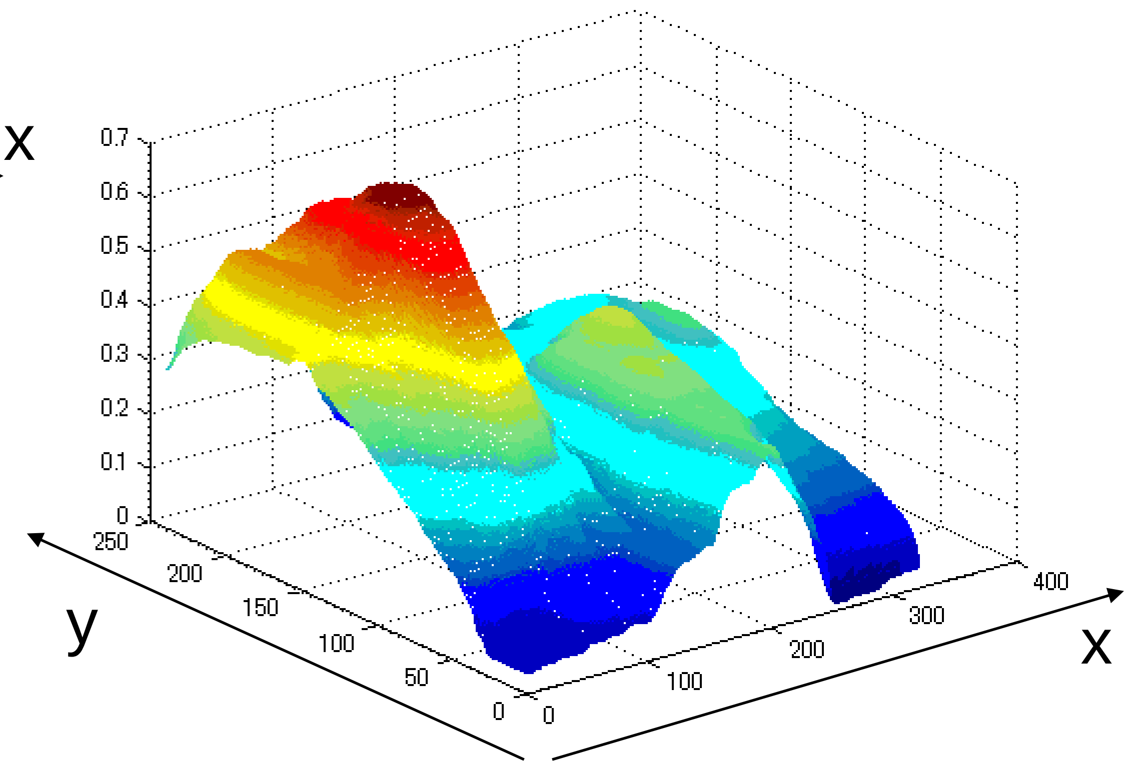
- Geometric interpretation: inner product of $(\sqrt{q_1}, \sqrt{q_2}, \cdots, \sqrt{q_m})^{\mathsf{T}}$ and $(\sqrt{p_1}, \sqrt{p_2}, \cdots, \sqrt{p_m})^{\mathsf{T}}$, which lie on the unit sphere surface

Similarity Map

current image



object model



Weighted Histogram

- The pixels near boundaries should have small influence
- Discontinuity in the similarity map is not favored

  ! **weight the voting depending of pixel locations**

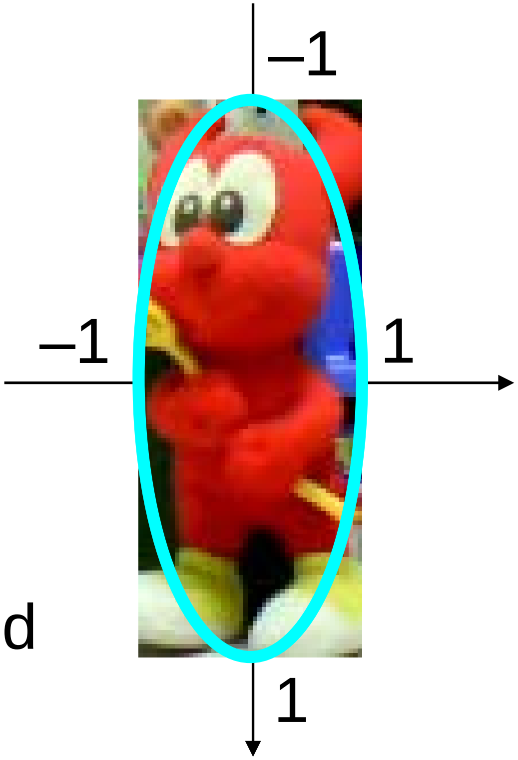
Object Model:

$$q_u \propto \sum_{x \in S_0(u)} k(\|\boldsymbol{x}\|^2)$$

$S_0(u)$: Set of pixels whose pixel values belong to bin u in the model image

k(): weight function or kernel function

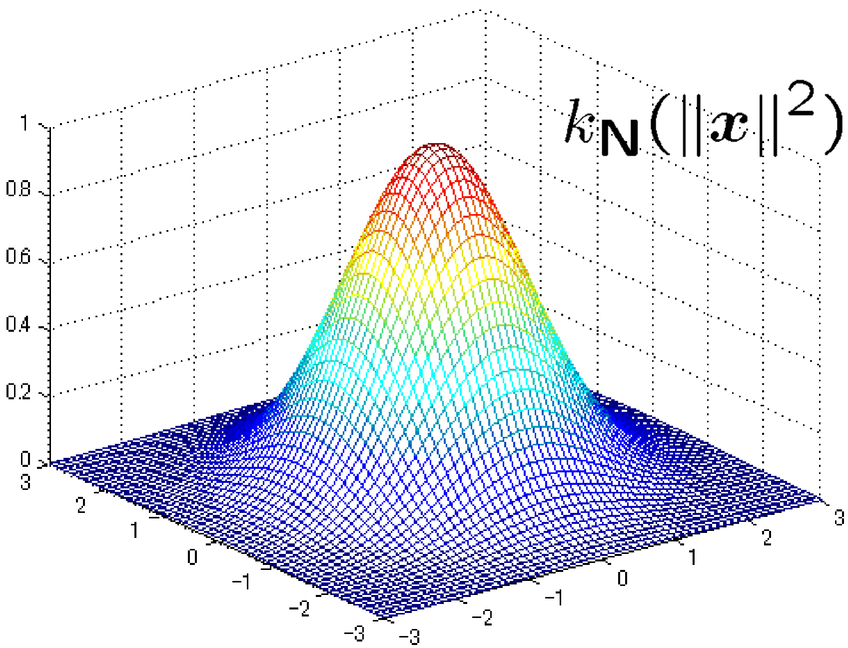
- centered at origin
- image coordinates **x** = (x, y) are normalized so that it fits the unit circle



Kernel Function Examples

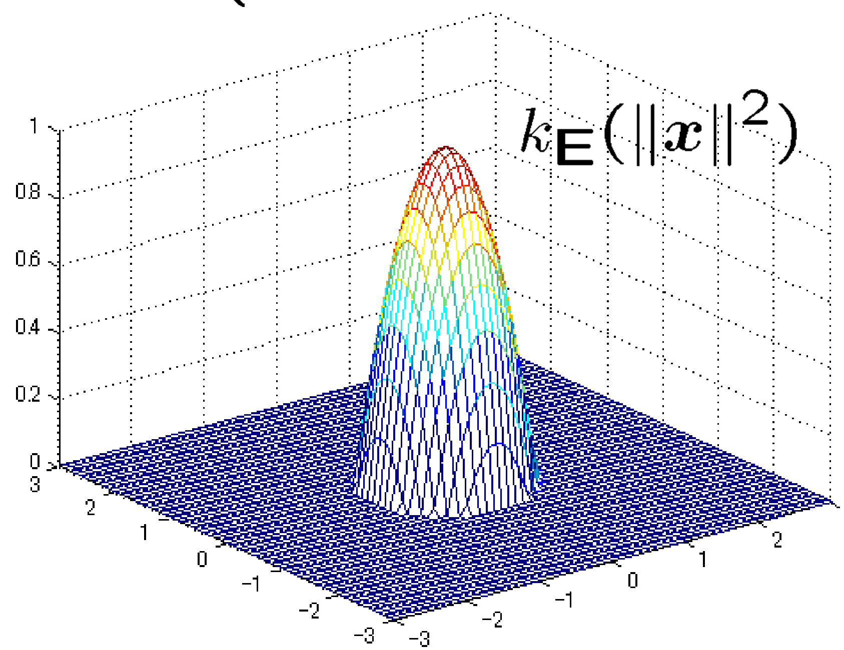
ex1) Gauss kernel

$$k_{\mathbf{N}}(x) \propto \exp(-\frac{x}{2})$$



ex2) Epanechnikov kernel
(kernel with Epanechnikov profile)

$$k_{\mathbf{E}}(x) \propto \begin{cases} 1 - x, & x \leq 1 \\ 0, & \textbf{otherwise} \end{cases}$$



Weighted Histogram of Candidate Region

Histogram of candidate region
(centered at **y**):

$$p_u(\boldsymbol{y}) \propto \sum_{\boldsymbol{x}\in S(u)} k(\|\boldsymbol{y}-\boldsymbol{x}\|^2)$$

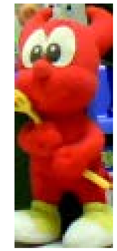
unit circle centered at **y**

S(u): Set of pixels whose pixel values belong to bin u in the current image

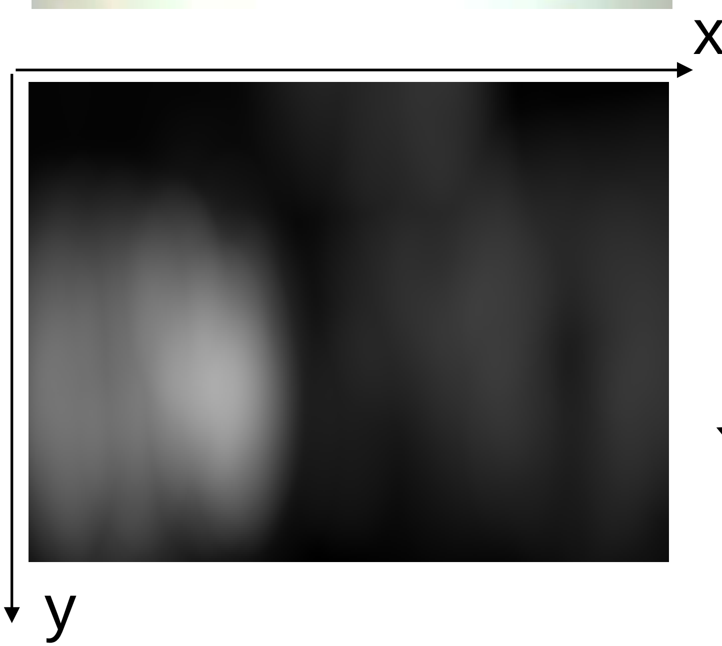
- Note again: image coordinates are normalized so that it fits the unit circle

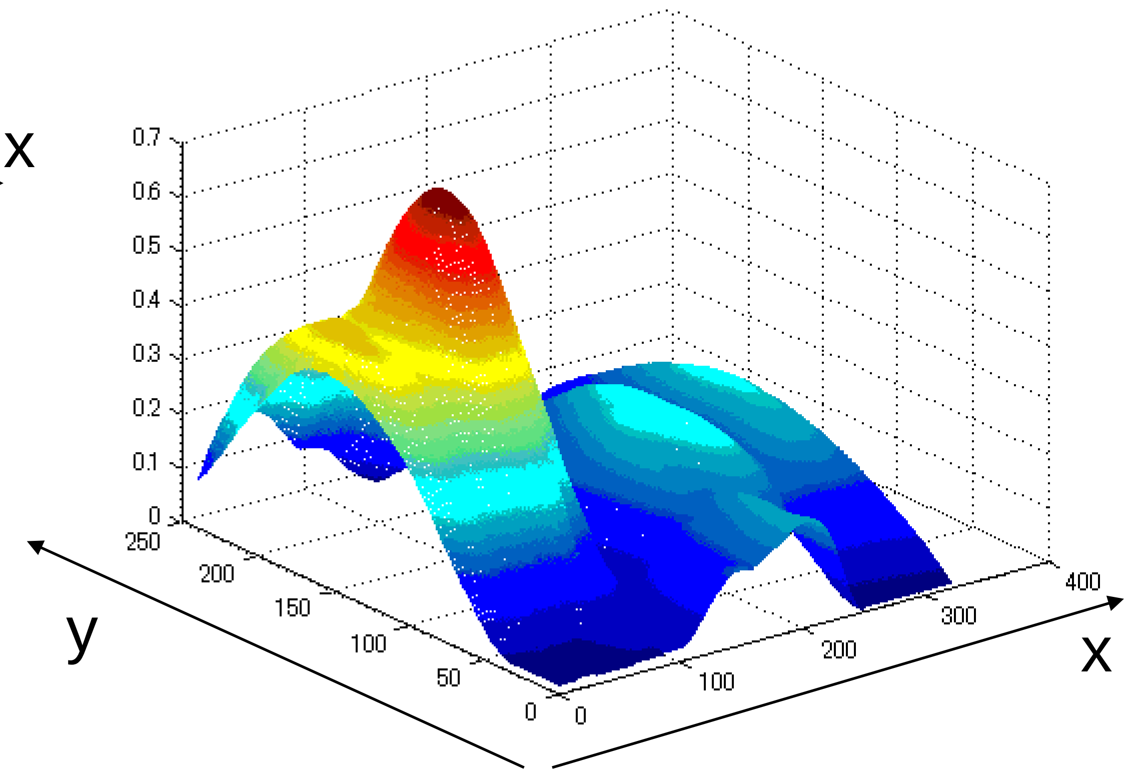
Similarity Map with Weighted Histogram

current image



object model





Approximating the Similarity

Since exhaustive search for maximum similarity is too time consuming, let's think of using a gradient-based method

- Let the initial candidate position be $\mathbf{y_0}$
- Consider 1st order Taylor expansion to ρ ($\mathbf{p}(\mathbf{y})$, $\mathbf{q}$) with respect to $\mathbf{p}(\mathbf{y})$ around $\mathbf{p}(\mathbf{y}_0)$

$$
\begin{aligned}
\rho(\boldsymbol{p}(\boldsymbol{y}), \boldsymbol{q}) &= \sum_u \sqrt{p_u(\boldsymbol{y})}\sqrt{q_u} \\
&\approx \sum_u \sqrt{q_u}\left\{\sqrt{p_u(\boldsymbol{y}_0)} + \frac{1}{2}p_u(\boldsymbol{y}_0)^{-1/2}\left(p_u(\boldsymbol{y}) - p_u(\boldsymbol{y}_0)\right)\right\} \\
&= \sum_u \sqrt{q_u}\left(\sqrt{p_u(\boldsymbol{y}_0)} + \frac{1}{2}p_u(\boldsymbol{y})\frac{1}{\sqrt{p_u(\boldsymbol{y}_0)}} - \frac{1}{2}\sqrt{p_u(\boldsymbol{y}_0)}\right) \\
&= \frac{1}{2}\sum_u \sqrt{q_u}\sqrt{p_u(\boldsymbol{y}_0)} + \frac{1}{2}\sum_u p_u(\boldsymbol{y})\frac{\sqrt{q_u}}{\sqrt{p_u(\boldsymbol{y}_0)}}
\end{aligned}
$$

Since the 1st term does not depend on y, what we should maximize is the 2nd term: $\sum_u p_u(\boldsymbol{y}) \frac{\sqrt{q_u}}{\sqrt{p_u(\boldsymbol{y}_0)}}$

Recalling that $p_u(\boldsymbol{y}) \propto \sum_{\boldsymbol{x} \in S(u)} k(\|\boldsymbol{y} - \boldsymbol{x}\|^2)$,

this comes down to maximization of

$$\sum_{u \in \textbf{all bins}} \sum_{x \in \textbf{all pixels belonging to } u} k(\|\boldsymbol{y} - \boldsymbol{x}\|^2) \frac{\sqrt{q_u}}{\sqrt{p_u(\boldsymbol{y}_0)}}$$

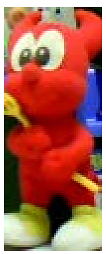
$$= \sum_{x \in \textbf{all pixels}} \sqrt{\frac{q_{b(\boldsymbol{x})}}{p_{b(\boldsymbol{x})}(\boldsymbol{y}_0)}} k(\|\boldsymbol{y} - \boldsymbol{x}\|^2)$$

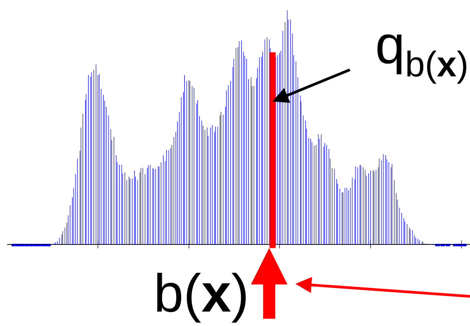
where b(**x**) is the bin to which **x** belongs

So, what we should maximize is:

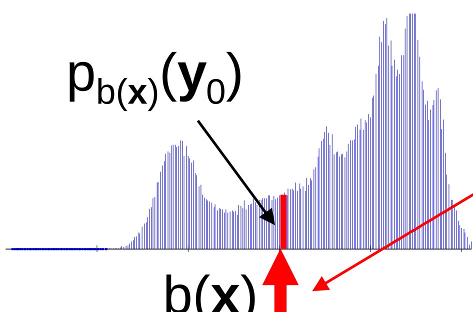
$$\sum_{\boldsymbol{x} \in \textbf{all pixels}} \sqrt{\frac{q_{b(\boldsymbol{x})}}{p_{b(\boldsymbol{x})}(\boldsymbol{y}_0)}}\, k(\|\boldsymbol{y}-\boldsymbol{x}\|^2) = \sum_{\boldsymbol{x} \in \textbf{all pixels}} w(\boldsymbol{x})\, k(\|\boldsymbol{y}-\boldsymbol{x}\|^2)$$

model histogram





histogram of region around $\mathbf{y}_0$



For each pixel **x** (in the kernel range), find b(**x**) to look up q and p, and compute w(x).

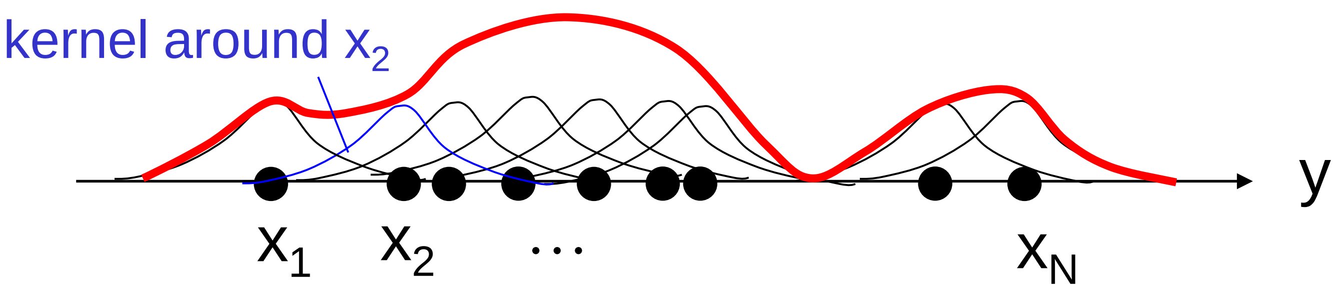
Kernel Density Estimation

$$\sum_{x \in \text{all pixels}} w(x)k(\|y-x\|^2)$$

- Our view until now: k() lies around **y** (and evaluated at each **x**, and summed over **x**)
- New view: k() is around each **x** (and evaluated at a single point **y**, and summed over **x**)

This new view is often used when one wants to estimate a probability distribution from finite samples drawn from the distribution

- called Kernel Density Estimation (KDE) or Parzen Estimation



Mean Shift Method

- An efficient method to find a local maximum of a probability distribution estimated by KDE

$$f_k(\boldsymbol{y}) = \sum_{\boldsymbol{x}} w(\boldsymbol{x}) k(\|\boldsymbol{y} - \boldsymbol{x}\|^2)$$

Gradient: $\nabla f_k(\boldsymbol{y}) = \frac{\partial}{\partial \boldsymbol{y}} f_k(\boldsymbol{y}) = \sum_{\boldsymbol{x}} k'(\|\boldsymbol{y} - \boldsymbol{x}\|^2) \cdot 2(\boldsymbol{y} - \boldsymbol{x}) w(\boldsymbol{x})$

Writing $g(x) = -k'(x)$, we have

$$\nabla f_k(\boldsymbol{y}) = 2 \sum_{\boldsymbol{x}} g(\|\boldsymbol{y} - \boldsymbol{x}\|^2)(\boldsymbol{x} - \boldsymbol{y}) w(\boldsymbol{x})$$

KDE with kernel g

$$= 2 \left[\sum_{\boldsymbol{x}} \left\{ \boldsymbol{x} w(\boldsymbol{x}) g(\|\boldsymbol{y} - \boldsymbol{x}\|^2) \right\} - \boldsymbol{y} \sum_{\boldsymbol{x}} \left\{ w(\boldsymbol{x}) g(\|\boldsymbol{y} - \boldsymbol{x}\|^2) \right\} \right]$$

$$= 2 f_g(\boldsymbol{y}) \left[\frac{\sum_{\boldsymbol{x}} \left\{ \boldsymbol{x} w(\boldsymbol{x}) g(\|\boldsymbol{y} - \boldsymbol{x}\|^2) \right\}}{f_g(\boldsymbol{y})} - \boldsymbol{y} \right]$$

Mean Shift vector

Mean Shift vector:

$$\boldsymbol{m}_g(\boldsymbol{y}) = \frac{\sum_{\boldsymbol{x}} \{\boldsymbol{x} w(\boldsymbol{x}) g(\|\boldsymbol{y}-\boldsymbol{x}\|^2)\}}{\sum_{\boldsymbol{x}} \{w(\boldsymbol{x}) g(\|\boldsymbol{y}-\boldsymbol{x}\|^2)\}} - \boldsymbol{y} = \frac{\nabla f_k(\boldsymbol{y})}{2 f_g(\boldsymbol{y})}$$

- toward the direction f_k(**y**) becomes larger
- large when f_g(**y**) is small, small when f_g(**y**) is large

As a special case, if you use Epanechnikov kernel as k(), g() becomes 1 within the unit circle, and 0 otherwise

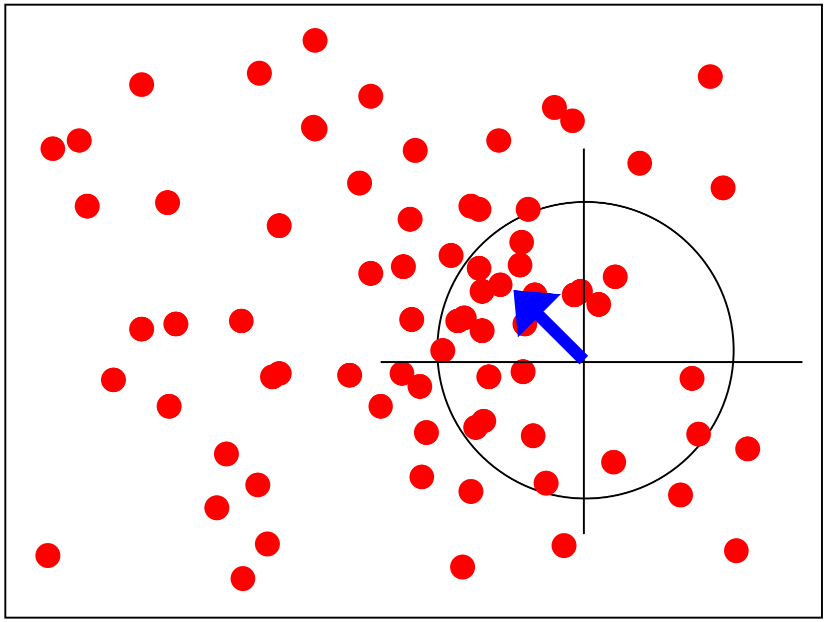
$$f_g(\boldsymbol{y}) = \sum_{\boldsymbol{x} \in \textbf{unit circle}} w(\boldsymbol{x})$$

center of gravity within unit circle

$$\boldsymbol{m}_g(\boldsymbol{y}) = \frac{\sum_{\boldsymbol{x} \in \textbf{unit circle}} \boldsymbol{x} w(\boldsymbol{x})}{\sum_{\boldsymbol{x} \in \textbf{unit circle}} w(\boldsymbol{x})} - \boldsymbol{y}$$

Mean Shift Method (general KDE problem)

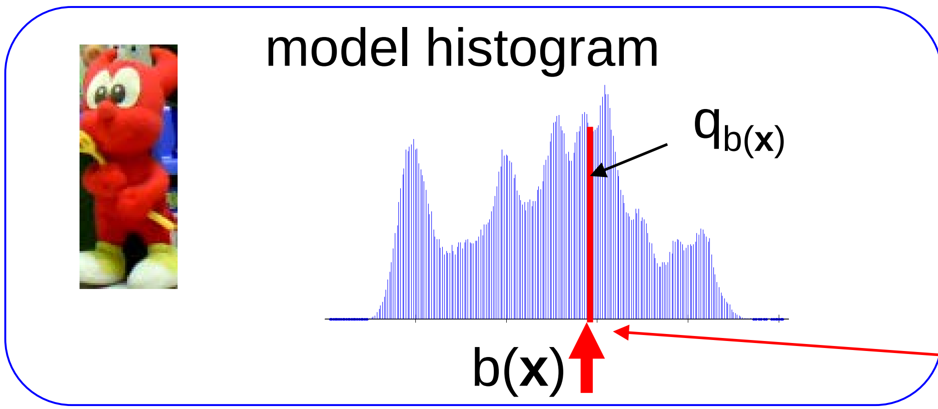
1. Compute center of gravity of samples around current position
2. Move to the center of gravity (Mean Shift)
3. Return to 1. unless the mean shift vector is too small

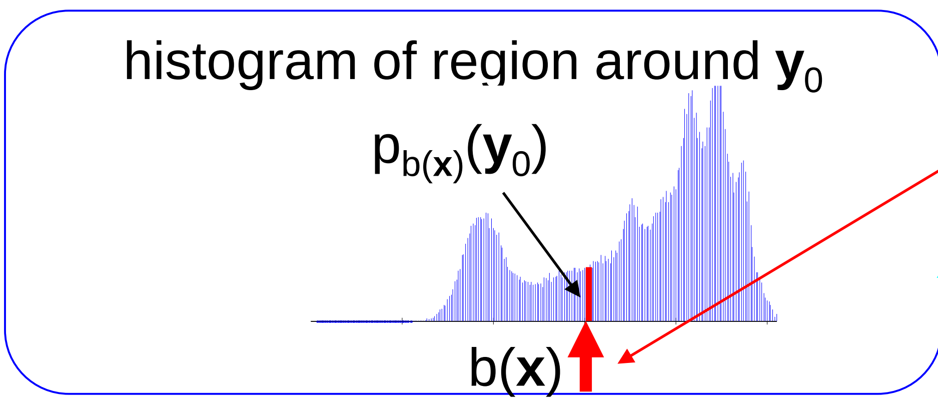


- When the maximum is far, $f_g(\mathbf{y})$ will be small and the shift will be large
- When the maximum is near, the shift will become small
- It has been proved that it converges to the local maximum with mild conditions for the shape of k()
  - Epanechnikov kernel satisfies it

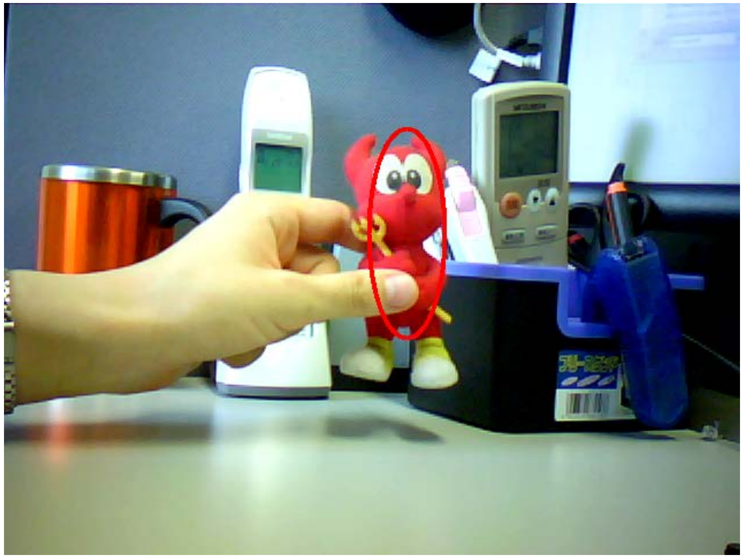
Mean Shift Method (for histogram tracking)

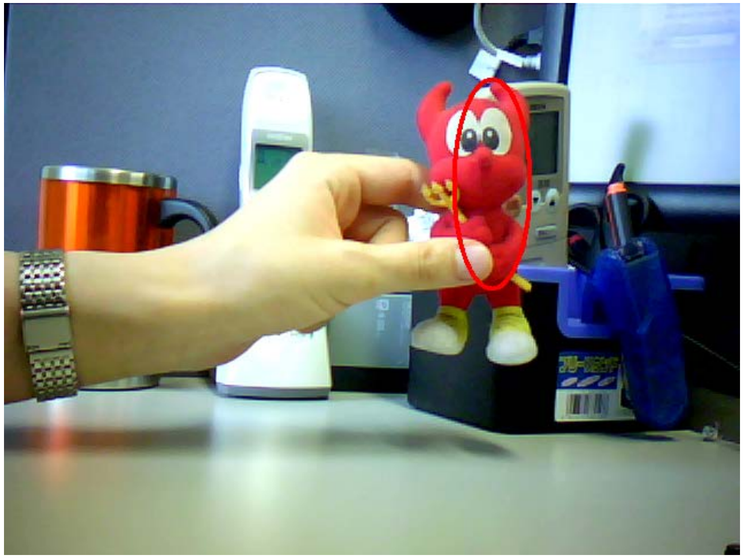
1. Compute the weighted histogram $p(\mathbf{y}_0)$ around $\mathbf{y}_0$
2. Move $\mathbf{y}_0$ to the center of gravity of $w(\mathbf{x})$, by finding $b(\mathbf{x})$ and looking up $\mathbf{q}$ an $\mathbf{p}$ for each pixel $\mathbf{x}$ around $\mathbf{y}_0$
3. Return to 1. unless the move is too small





$$w(\boldsymbol{x}) = \sqrt{\frac{q_{b(\boldsymbol{x})}}{p_{b(\boldsymbol{x})}(\boldsymbol{y}_0)}}$$





References

- D. Comaniciu, V. Ramesh and P. Meer: Kernel-Based Object Tracking, IEEE Trans. of Pattern Analysis and Machine Intelligence, vol.25, no.5, 2003.
- D. Comaniciu and P. Meer: Mean Shift: A Robust Approach Toward Feature Space Analysis, IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.25, no.5, 2003.
- K. Fukunaga and L. D. Hostetler: The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition, IEEE Trans. on Information Theory, vol.IT-21, no.1, 1975.

Assignments

1. Describe your affiliation (e.g. department, laboratory name) and your prospective research topic.
2. Write an image processing program with your favorite programming language, and report the results with brief explanations. Attachment of the source codes are unnecessary (unless needed for explanation).
   - Just running the sample codes is not acknowledged
   - Re-implementing or improving the sample codes are OK
3. Rewrite the above program in some respects, and discuss the changes of the results and/or the speeds.
   - e.g.: See the results with respect to some design parameters
4. Describe your impression of the topics of lectures (by Kagami) and these assignments.
   - already known or unknown?; difficult or easy?; good points / bad points?; suggested improvements?

Due: August 9 (Thu.) @ Mech. Bldg. 2, Room 419 (or E-mail me).
Include your Name, Student ID, E-mail address.

(Ask me for help in advance if you have any difficulty in programming)