KLT Tracker

1. Detect Harris corners in the first frame
2. For each Harris corner compute motion between consecutive frames (Alignment).
3. Link motion vectors in successive frames to get a track
4. Introduce new Harris points at every $m$ frames
5. Track new and old Harris points using steps 2-4.
Mean-Shift Tracking

Lecture-11
Mean-Shift Tracking
Mean-Shift Tracking
Mean-Shift Tracking
Presentations

- Comaniciu et al
- Alper Yilmaz
- Afshin Dehghan
Mean-Shift Theory and Its Applications

Lecture-18

Objective: Find the densest region
Objective: Find the densest region
Distribution of identical billiard balls
Objective: Find the densest region

Distribution of identical billiard balls
Objective: Find the densest region
Distribution of identical billiard balls
Objective: Find the densest region
Distribution of identical billiard balls
Objective: Find the densest region
Distribution of identical billiard balls
Objective: Find the densest region
Distribution of identical billiard balls
Objective: Find the densest region

Distribution of identical billiard balls
Mean Shift Vector

Given:
Data points and approximate location of the mean of this data:

Task:
Estimate the exact location of the mean of the data by determining the shift vector from the initial mean.
Mean Shift Vector Example

\[ M_h(y) = \left[ \frac{1}{n_x} \sum_{i=1}^{n_x} x_i \right] - y_0 \]

Mean shift vector always points towards the direction of the maximum increase in the density.
Mean Shift (Weighted)

\[
M_h(y_0) = \frac{\sum_{i=1}^{n_x} w_i(y_0) x_i}{\sum_{i=1}^{n_x} w_i(y_0)} - y_0
\]

- \(n_x\) : number of points in the kernel
- \(y_0\) : initial mean location
- \(x_i\) : data points
- \(h\) : kernel radius

Weights are determined using kernels (masks):
Uniform, Gaussian or Epanechnikov
Properties of Mean Shift

• Mean shift vector has the direction of the gradient of the density estimate.

• It is computed iteratively for obtaining the maximum density in the local neighborhood.
What is Mean-Shift?

- A tool for finding modes in a set of data samples, manifesting an underlying probability density function (PDF) in $\mathbb{R}_N$
Non-Parametric Density Estimation

- Assumption: The data points are sampled from an underlying PDF

Data point density implies PDF value!
Non-Parametric Density Estimation

Assumed Underlying PDF

Real Data Samples
Non-Parametric Density Estimation

Assumed Underlying PDF

Real Data Samples
Parametric Density Estimation

- Assumption: The data points are sampled from an underlying PDF

\[
\text{PDF}(x) = \sum_{i} c_i \cdot e^{-\frac{(x-u_i)^2}{2\sigma_i^2}}
\]

Assumed Underlying PDF

Real Data Samples
Kernel Density Estimation
Various Kernels

\[ P(x) = \frac{1}{n} \sum_{i=1}^{n} K(x - x_i) \]

A function of some finite number of data points \( x_1 \ldots x_n \)

Examples:

- **Epanechnikov Kernel**
  \[ K_E(x) = \begin{cases} 
  c \left(1 - \|x\|^2\right) & \|x\| \leq 1 \\
  0 & \text{otherwise} 
\end{cases} \]

- **Uniform Kernel**
  \[ K_U(x) = \begin{cases} 
  c & \|x\| \leq 1 \\
  0 & \text{otherwise} 
\end{cases} \]

- **Normal Kernel**
  \[ K_N(x) = c \cdot \exp\left(-\frac{1}{2}\|x\|^2\right) \]
Profile and Kernel

Radially symmetric Kernel

\[ K(x) = ck(||x||^2) \]

Profile

\[ P(x) = \frac{1}{n} \sum_{i=1}^{n} K(x - x_i) = \frac{1}{n} c \sum_{i=1}^{n} k(||x - x_i||^2) \]
Kernel Density Estimation

\[ P(x) = \frac{1}{n} c \sum_{i=1}^{n} k(||x - x_i||^2) \]

\[ \nabla P(x) = \frac{1}{n} c \sum_{i=1}^{n} \nabla k(||x - x_i||^2) \]

\[ \nabla P(x) = -\frac{2}{n} c \sum_{i=1}^{n} (x - x_i)k'(||x - x_i||^2) \]
Kernel Density Estimation

\[ \nabla P(x) = \frac{1}{n} 2c \sum_{i=1}^{n} (x - x_i)k'(\|x - x_i\|^2) \]

\[ \nabla P(x) = \frac{1}{n} 2c \sum_{i=1}^{n} (x_i - x)g(\|x - x_i\|^2) \]

\[ \nabla P(x) = \frac{1}{n} 2c \sum_{i=1}^{n} x_i g(\|x - x_i\|^2) - \frac{1}{n} 2c \sum_{i=1}^{n} x g(\|x - x_i\|^2) \]

\[ \nabla P(x) = \frac{1}{n} 2c \sum_{i=1}^{n} g(\|x - x_i\|^2) \left[ \frac{1}{n} \sum_{i=1}^{n} x_i g(\|x - x_i\|^2) \right] - \frac{1}{n} 2c \sum_{i=1}^{n} g(\|x - x_i\|^2) \]

\[ g(x) = k'(x) \]
\[ \nabla P(x) = \frac{1}{n} 2c \sum_{i=1}^{n} g(\|x - x_i\|^2) \left[ \frac{\sum_{i=1}^{n} x_i g(\|x - x_i\|^2)}{\sum_{i=1}^{n} g(\|x - x_i\|^2)} - x \right] \]
Computing The Mean Shift

\[ \nabla P(x) = \frac{c}{n} \sum_{i=1}^{n} \nabla k_i = \frac{c}{n} \left( \sum_{i=1}^{n} g_i \right) \]

\[ \nabla P(x) = \frac{c}{n} \sum_{i=1}^{n} g_i \times m(x) \]

\[ m(x) = \frac{\nabla P(x)}{\frac{c}{n} \sum_{i=1}^{n} g_i} \]
Updated Mean Shift Procedure:
• Find all modes using the Simple Mean Shift Procedure
• Prune modes by perturbing them (find saddle points and plateaus)
• Prune nearby – take highest mode in the window

What happens if we reach a saddle point?

Perturb the mode position and check if we return back
Mean Shift Properties

- Automatic convergence speed
  - mean shift vector size depends on gradient.

- Near maxima, the steps are small and refined

- Convergence is guaranteed for infinitesimal steps only, (therefore set a lower bound)

- For Uniform Kernel ( ), convergence is achieved in a finite number of steps

- Normal Kernel ( exhibits a smooth trajectory, but is slower than Uniform Kernel ( ).
Real Modality Analysis

Tessellate the space with windows
Run the procedure in parallel
The blue data points were traversed by the windows towards the mode.
Mean Shift Applications
Mean-Shift Object Tracking

- General Framework

Choose the model in the initial frame → Choose a feature space → Represent the model in the selected feature space

- The object is being modeled using color probability density
Mean-Shift Object Tracking

- General framework

Select a ROI around the target location in current frame

Target Localization-Tracking

Find the most similar candidate based on the similarity func
Mean-Shift Object Tracking

- PDF Representation

**Target Model**: color distribution by discrete m-bin color histogram

\[ \tilde{q} = \{q_u\}_{u=1}^{m}, \quad \sum_{u=1}^{m} q_u = 1 \]

**Current Frame**

**Next Frame**

**Candidate Model**: color distribution by discrete m-bin color histogram

\[ p(y) = \{p_u(y)\}_{u=1}^{m}, \quad \sum_{u=1}^{m} p_u = 1 \]

**Similarity Function**

\[ f(y) = f[\tilde{q}, \overline{p(y)}] \]

**The Bhattacharyya Coefficient**
Mean-Shift Object Tracking

• The Bhattacharyya Coefficient
  • Measures similarity between object model \( q \) and color \( p \) of target at location \( y \)

\[
\rho(p(y), q) = \sum_{u=1}^{m} \sqrt{p_u(y)q_u}
\]

• \( \rho \) is the cosine of vectors \((\sqrt{p_1}, ..., \sqrt{p_m})^T\) and \((\sqrt{q_1}, ..., \sqrt{q_m})^T\).
• Large \( \rho \) means good match between candidate and target model
• In order to find the new target location we try to maximize the Bhattacharyya coefficient
Target Model for Tracking

• Features used for tracking include:
  • Gray level
  • Color
  • Gradient

• Feature probability distribution are calculated by using weighted histograms.

• The weights are derived from Epanechnikov profile.
Distribution

$x_1, x_2, x_3, x_4$ have the same feature, such as gray level.

$$p(u) = C \sum_{x_i \in S} k\left(\|x_i\|^2\right) \delta\left[ S(x_i) - u \right]$$

$S(x_i)$ is the color at $x_i$
Target Gray Level Feature

- Target 1
- Target 2
- Non-target

Target 1 distribution
Target 2 distribution
Non-target distribution

Image histogram
Similarity of Target and Candidate Distributions

Target : \( q_u \).
Candidate : \( p_u \).

\[
d(y) = \sqrt{1 - \rho(y)}
\]

\[
\rho(y) = \rho[\hat{p}(y), q] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)q_u}
\]

\( \rho(y) \) : Bhattacharya coefficient.
**Distance Minimization**

Minimizing the distance corresponds to maximizing Bhattacharya coefficient.

\[
\rho[\hat{p}(y), q] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)} q_u
\]

Taylor expansion around \(\hat{p}(y_0)\)

\[
\rho[\hat{p}(y), q] \approx \rho[\hat{p}(y_0), q] + \frac{1}{2} \sum_{i=1}^{m} \hat{p}_u(y) \sqrt{\frac{q_u}{\hat{p}_u(y_0)}}
\]

Maximizing Bhattacharya coefficient can be obtained by maximizing the blue term.
Likelihood Maximization

\[ \rho[\hat{p}(y), q] = \rho[\hat{p}(y_0), q] + \frac{1}{2} \sum_{i=1}^{m} \hat{p}_u(y) \sqrt{\frac{q_u}{\hat{p}_u(y_0)}} \]

\[ \frac{C_h}{2} \sum_{i=1}^{n_x} \left[ \sum_{u=1}^{m} \delta[S(x_i) - u] \sqrt{\frac{q_u}{\hat{p}_u(y_0)}} \right] k\left(\frac{\|y - x_i\|}{h}\right) \]

- \( h \): radius of sphere
- \( C_h \): normalization constant
- \( S(x_i) \): gray level at \( x \)
- \( y \): kernel center
- \( m \): number of bins

likelihood maximization depends on maximizing \( w_i \).
Likelihood Maximization
Using Mean Shift Vector

Maximization of the likelihood of target and candidate depends on the weights:

\[ w_i(y_o) = \sum_{u=1}^{m} \delta[S(x_i) - u] \sqrt{\frac{q_u}{\hat{P}_u(y_o)}} \quad \text{where} \quad 0 \leq w_i \leq 1 \]

Since \( \sum_{i=1}^{n_x} w_i(y_0) \) is strictly positive, mean shift vector can be written as

\[ M_h(y_0) = \frac{\sum_{i=1}^{n_x} w_i(y_0)x_i}{\sum_{i=1}^{n_x} w_i(y_0)} - y_0 \]

Thus, new target center is

\[ \hat{y} = y_0 + M_h(y_0) \]
Algorithm

1. Calculate \( q \)
2. Initialize estimated center \( y_1 = y_0 \)
3. Calculate \( p \)
4. Calculate \( w \)
5. Estimate new target center \( y_1 \)
6. \( d < \varepsilon \)
7. Update target center \( y_0 = y_1 \)
8. Repeat until end of the sequence
Tracking A Single Point
References
