Visual Tracking

Introduction to Computer Vision
CSE 152
Lecture 13

Announcements

- Homework 3 is due May 18, 11:59 PM
- Reading:
  – Chapter 11: Tracking

Visual Tracking

Main Challenges

1. 3D pose variation
2. Target occlusion
3. Illumination variation
4. Camera jitter
5. Expression variation
   etc.

[Ho, Lee, Kriegman]

Main tracking notions

- State: usually a finite number of parameters (a vector) that characterizes the "state" (e.g., location, size, pose, deformation) of the object being tracked.
- Dynamics: How does the state change over time? How is that changed constrained?
- Representation: How do you represent the object being tracked?
- Prediction: Given the state at time *t-1*, what is an estimate of the state at time *t*?
- Correction: Given the predicted state at time *t* and a measurement at time *t*, update the state.
- Initialization: What is the state at time *t* = 0?

What is the state?

- 2D image location $\Phi=(u,v)$
- Image location + scale $\Phi=(u,v,s)$
- Image location + scale + orientation $\Phi=(u,v,s,\theta)$
- Affine transformation
- 3D pose
- 3D pose plus internal shape parameters (some may be discrete)
  – e.g., for a face, 3D pose + facial expression using FACS + eye state (open/closed)
- Collections of control points specifying a spline
- Above, but for multiple objects (e.g., tracking a formation of airplanes)
- Augment above with temporal derivatives $(\phi, \dot{\phi})$

State Examples

– Object is ball, state is 3D position + velocity, measurements are derived from stereo pairs
– Object is person, state is body configuration, measurements are derived from video frames
– What is state here?

Example: Blob Tracker

- From input image *I(u,v)* at time *t*, create a binary image by applying a function f(I(u,v))
- Clean up binary image using morphological operators
- Perform connected component exploration to find “blobs” (i.e., connected regions)
- Compute their moments (mean and covariance of region coordinates) and use as state
- Using state estimate from time *t-1* and perform “data association” to identify state at time *t*

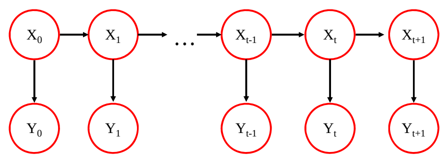
Blob Tracking in IR Images

- Threshold about body temperature
- Connected component analysis
- Position, scale, orientation of regions
- Temporal coherence

Tracking: Probabilistic framework

- Very general model
  - Assume there are moving objects that have an underlying state X
  - There are observations (measurements) Y, some of which are functions of this state
  - Over time
    - The state changes: X_{t-1}, X_t, X_{t+1}
    - There are new observations: Y_{t-1}, Y_t, Y_{t+1}

Tracking State



- Instead of “knowing state” at each instant, we treat the state as random variables X_t characterized by a pdf $P(X_t)$ or perhaps conditioned on other random variables, e.g., $P(X_t \mid X_{t-1})$ etc.
- The observation (measurement) Y_t is a random variable conditioned on the state $P(Y_t \mid X_t)$
- Generally, we don’t observe the state – it’s hidden

Three main steps

- **Prediction:** we have seen $\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}$ — what state does this set of measurements predict for the i'th frame? to solve this problem, we need to obtain a representation of $P(\boldsymbol{X}_i|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{i-1} = \boldsymbol{y}_{i-1})$.
- **Data association:** Some of the measurements obtained from the i-th frame may tell us about the object's state. Typically, we use $P(\boldsymbol{X}_i|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{i-1} = \boldsymbol{y}_{i-1})$ to identify these measurements.
- **Correction:** now that we have $\boldsymbol{y}_i$ — the relevant measurements — we need to compute a representation of $P(\boldsymbol{X}_i|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_i = \boldsymbol{y}_i)$.

We can try to express these conditional distributions parametrically, sample the distribution, or estimate the mode.

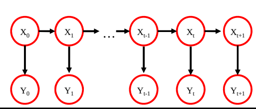
Simplifying Assumptions

- **Only the immediate past matters:** formally, we require

$$P(\boldsymbol{X}_i|\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{i-1}) = P(\boldsymbol{X}_i|\boldsymbol{X}_{i-1})$$

- **Measurements depend only on the current state:** we assume that $\boldsymbol{Y}_i$ is conditionally independent of all other measurements given $\boldsymbol{X}_i$. This means that

$$P(\mathbf{Y_i} \mid \mathbf{Y}_1, \ldots, \mathbf{Y_{i-1}}, \mathbf{X_i}) = P(\mathbf{Y_i} \mid \mathbf{X_i})$$



Tracking as induction

- Assume data association is done
  - Sometimes challenging in cluttered scenes. See work by Christopher Rasmussen on Joint Probabilistic Data Association Filters (JPDAF).
- Do correction for frame $i = 0$
- Assume we have corrected estimate for frame i
  - We can prediction the estimate for frame $i + 1$, correction for frame $i + 1$

Base case

P(y | x) is our observation model.
For example, P(y | x) might be a Gaussian with mean x.

Firstly, we assume that we have $P(\boldsymbol{X}_0)$

Prior distribution of initial state

And, we make a measurement $\mathbf{y_0}$

$$
\begin{aligned}
P(\boldsymbol{X}_0 | \boldsymbol{Y}_0 = \boldsymbol{y}_0) &= \frac{P(\boldsymbol{y}_0 | \boldsymbol{X}_0) P(\boldsymbol{X}_0)}{P(\boldsymbol{y}_0)} \\
&= \frac{P(\boldsymbol{y}_0 | \boldsymbol{X}_0) P(\boldsymbol{X}_0)}{\int P(\boldsymbol{y}_0 | \boldsymbol{X}_0) P(\boldsymbol{X}_0) d\boldsymbol{X}_0} \\
&\propto P(\boldsymbol{y}_0 | \boldsymbol{X}_0) P(\boldsymbol{X}_0)
\end{aligned}
$$

Induction step: State Prediction

Given $P(\boldsymbol{X}_{i-1} | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})$.

Prediction

Prediction involves representing

$$P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})$$

Our independence assumptions make it possible to write

$$
\begin{aligned}
P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) &= \int P(\boldsymbol{X}_i, \boldsymbol{X}_{i-1} | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) d\boldsymbol{X}_{i-1} \\
&= \int P(\boldsymbol{X}_i | \boldsymbol{X}_{i-1}, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) P(\boldsymbol{X}_{i-1} | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) d\boldsymbol{X}_{i-1} \\
&= \int P(\boldsymbol{X}_i | \boldsymbol{X}_{i-1}) P(\boldsymbol{X}_{i-1} | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) d\boldsymbol{X}_{i-1}
\end{aligned}
$$

Induction step: State Correction

In prediction, we estimated the state X_i given the measurements up to i-1.
Now we get the measure at time i called y_i.

Correction

Correction involves obtaining a representation of

$$P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)$$

Our independence assumptions make it possible to write

$$
\begin{aligned}
P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_i) &= \frac{P(\boldsymbol{X}_i, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)}{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)} \\
&= \frac{P(\boldsymbol{y}_i | \boldsymbol{X}_i, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})}{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)} \\
&= P(\boldsymbol{y}_i | \boldsymbol{X}_i) P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) \frac{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})}{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)} \\
&= \frac{P(\boldsymbol{y}_i | \boldsymbol{X}_i) P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})}{\int P(\boldsymbol{y}_i | \boldsymbol{X}_i) P(\boldsymbol{X}_i | \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) d\boldsymbol{X}_i}
\end{aligned}
$$

How is this formulation used

1. It's ignored. At each time instant, the state is estimated (perhaps a maximum likelihood estimate or something non-probabilistic).
2. The conditional distributions are represented by some convenient parametric form (e.g., Gaussian).
3. The PDFs are represented non-parametrically, and sampling techniques are used.

Linear dynamic models

- Use notation ~ to mean "has the pdf of," N(a, B) is a normal distribution with mean a and covariance B.
- A linear dynamic model has the form

$$
\begin{aligned}
\mathbf{x}_i &= N\left(\mathbf{D}_{i-1}\mathbf{x}_{i-1}; \Sigma_{d_i}\right) \\
\mathbf{y}_i &= N\left(\mathbf{M}_i\mathbf{x}_i; \Sigma_{m_i}\right)
\end{aligned}
$$

Examples

- Points moving with constant velocity
- Points moving with constant acceleration
- Periodic motion
- Etc.

CSE 152, Spring 2016 Introduction to Computer Vision

Points moving with constant velocity

- We have

$$u_i = u_{i-1} + \Delta t v_{i-1} + \varepsilon_i \quad \text{Position}$$
$$v_i = v_{i-1} + \varsigma_i \quad \text{Velocity}$$

  – (the Greek letters denote noise terms)
- Stack (u, v) into a single state vector

$$\begin{pmatrix} u \\ v \end{pmatrix}_i = \underbrace{\begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix}}_{D_{i-1}} \begin{pmatrix} u \\ v \end{pmatrix}_{i-1} + \text{noise}$$

which is the form we had above

CSE 152, Spring 2016 Introduction to Computer Vision

Points moving with constant acceleration

- We have

$$u_i = u_{i-1} + \Delta t v_{i-1} + \varepsilon_i$$
$$v_i = v_{i-1} + \Delta t a_{i-1} + \varsigma_i$$
$$a_i = a_{i-1} + \xi_i$$

  – (the Greek letters denote noise terms)
- Stack (u, v) into a single state vector

$$\begin{pmatrix} u \\ v \\ a \end{pmatrix}_i = \begin{pmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ a \end{pmatrix}_{i-1} + \text{noise}$$

which is the form we had above

CSE 152, Spring 2016 Introduction to Computer Vision

The Kalman Filter

- Key ideas:
  - Linear models interact uniquely well with Gaussian noise
    - Make the prior Gaussian, everything else Gaussian and the calculations are easy
  - Gaussians are really easy to represent
    - mean vector
    - covariance matrix

CSE 152, Spring 2016 Introduction to Computer Vision

The Kalman Filter in 1D

- Dynamic Model

$$x_i \sim N(d_i x_{i-1}, \sigma_{d_i}^2)$$

$$y_i \sim N(m_i x_i, \sigma_{m_i}^2)$$

- Notation

mean of $P(X_i|y_0, \ldots, y_{i-1})$ as $\overline{X}_i^-$ — Predicted mean

Corrected mean — mean of $P(X_i|y_0, \ldots, y_i)$ as $\overline{X}_i^+$

the standard deviation of $P(X_i|y_0, \ldots, y_{i-1})$ as σ_i^-

of $P(X_i|y_0, \ldots, y_i)$ as σ_i^+

CSE 152, Spring 2016 Introduction to Computer Vision

Prediction for 1-D Kalman filter

- The new state is obtained by
  - multiplying old state by known constant
  - adding zero-mean noise
- Therefore, predicted mean for new state is
  - constant times mean of old state
- Predicted variance is
  - sum of constant^2 times old state variance and noise variance

Because:
- Old state is normal random variable,
- Multiplying normal random variable by constant implies
  - mean is multiplied by a constant
  - variance is multiplied by square of constant
- Adding zero mean noise adds zero to the mean,
- Adding random variables adds variance

CSE 152, Spring 2016 Introduction to Computer Vision

Dynamic Model:

$$x_i \sim N(d_i x_{i-1}, \sigma_{d_i})$$

$$y_i \sim N(m_i x_i, \sigma_{m_i})$$

Start Assumptions: $\overline{x}_0^-$ and σ_0^- are known
Update Equations: Prediction

$$\boxed{\begin{aligned} \overline{x}_i^- &= d_i \overline{x}_{i-1}^+ \\ \sigma_i^- &= \sqrt{\sigma_{d_i}^2 + (d_i \sigma_{i-1}^+)^2} \end{aligned}}$$

Update Equations: Correction

$$x_i^+ = \left(\frac{\overline{x}_i^- \sigma_{m_i}^2 + m_i y_i (\sigma_i^-)^2}{\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2} \right)$$

$$\sigma_i^+ = \sqrt{\left(\frac{\sigma_{m_i}^2 (\sigma_i^-)^2}{(\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2)} \right)}$$

Correction for 1D Kalman filter

- Notice:
  - if measurement noise is small, then we rely mainly on the measurement
  - if measurment noise is large, then we rely mainly on the prediction

$$x_i^+ = \left(\frac{\overline{x}_i^- \sigma_{m_i}^2 + m_i y_i (\sigma_i^-)^2}{\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2} \right)$$

$$\sigma_i^+ = \sqrt{\left(\frac{\sigma_{m_i}^2 (\sigma_i^-)^2}{(\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2)} \right)}$$

Multivariate Kalman Filter

Dynamic Model:

$$\boldsymbol{x}_i \sim N(\mathcal{D}_i \boldsymbol{x}_{i-1}, \Sigma_{d_i})$$

$$\boldsymbol{y}_i \sim N(\mathcal{M}_i \boldsymbol{x}_i, \Sigma_{m_i})$$

Start Assumptions: $\overline{\boldsymbol{x}}_0^-$ and Σ_0^- are known
Update Equations: Prediction

$$\overline{\boldsymbol{x}}_i^- = \mathcal{D}_i \overline{\boldsymbol{x}}_{i-1}^+$$

$$\Sigma_i^- = \Sigma_{d_i} + \mathcal{D}_i \sigma_{i-1}^+ \mathcal{D}_i$$

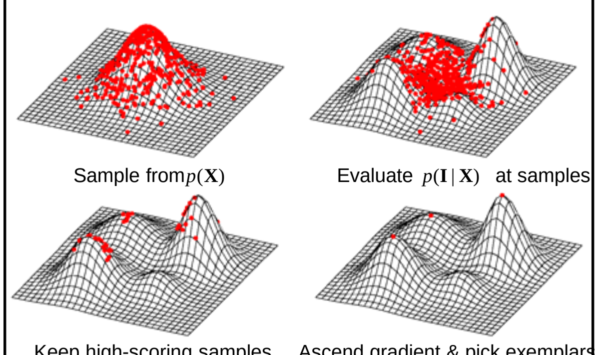
Update Equations: Correction

$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i} \right]^{-1}$$

$$\overline{\boldsymbol{x}}_i^+ = \overline{\boldsymbol{x}}_i^- + \mathcal{K}_i \left[\boldsymbol{y}_i - \mathcal{M}_i \overline{\boldsymbol{x}}_i^- \right]$$

$$\Sigma_i^+ = [Id - \mathcal{K}_i \mathcal{M}_i] \Sigma_i^-$$

Another Approach: Measurement Generation



Tracking Modalities

(Define the features (observations, measurements) Y_i)

- Color
  - Histogram [Birchfield 1998; Bradski 1998]
  - Volume [Wren *et al*., 1995; Bregler, 1997; Darrell, 1998]
- Shape
  - Deformable curve [Kass *et al*.1988]
  - Template[Blake *etal*. 1993; Birchfield 1998]
  - Example-based [Cootes *et al*., 1993; Baumberg & Hogg, 1994]
- Appearance
  - Correlation [Lucas & Kanade, 1981; Shi & Tomasi, 1994]
  - Photometric variation [Hager & Belhumeur, 1998]
  - Outliers [Black *et al*., 1998; Hager & Belhumeur, 1998]
  - Nonrigidity [Black *et al*., 1998; Sclaroff & Isidoro, 1998]
- Motion
  - Background model [Wren *et al*., 1995; Rosales & Sclaroff, 1999; Stauffer & Grimson, 1999]
  - Optical flow [Cutler & Turk]
  - Egomotion [Sawhney & Ayer, 1996; Irani & Anandan, 1998]
- Stereo
  - Blob correlation [Azarbayejani & Pentland, 1996]
  - Disparity map [Kanade *et al*., 1996; Konolige, 1997; Darrell *et al*., 1998]

Color Blob tracking

- **Color-based tracker gets lost on white knight: Same Color**

Snakes: Active Contours

- Contour C: continuous curve on smooth surface in $\Re^3$
- Snake S: projection of C to image
- Curve types
  - Edge between regions on surface with contrasting properties
  - Line that contrasts with surface properties on both side
  - Silhouette of surface against contrasting background
- General Algorithm:
  - Perform edge detection
  - Fit parametric or non-parametric curve to data

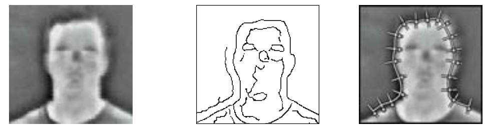
Snakes: Basic Approach

- Parameterize a closed contour

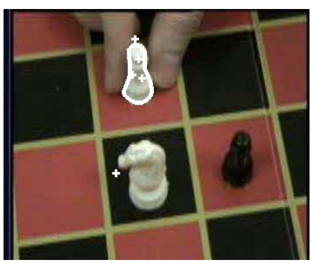
$$\mathbf{Q} = (q_0^x .. q_n^x, q_0^y .. q_n^y)$$

$$\mathbf{U}(s) = \begin{pmatrix} \mathbf{B}(s)^t & 0 \\ 0 & \mathbf{B}(s)^t \end{pmatrix}$$

- $\mathbf{r}(s) = \mathbf{q}^t\mathbf{B}(s)$ or $\mathbf{r}(s) = \mathbf{U}(s)\mathbf{Q}$
- Given a predicted state **q,** search radially for edges
- Solve a least squares problem for new state

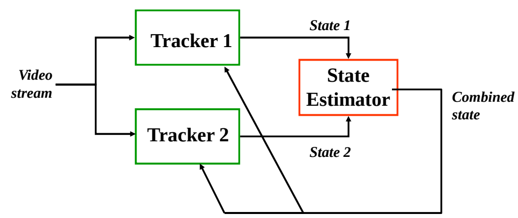


Tracker Composition: Only Shape (Snakes)



- **Geometry-based tracker gets lost on black pawn: Same shape**

Tracker Composition

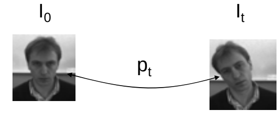


Tracker Composition: Color and Shape

- **Combining Trackers => Robustness**

Visual Tracking using regions

I_0 I_t



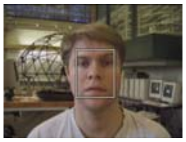
Variability model: $I_t = g(I_0, p_t)$

Incremental Estimation: From I_0, I_{t+1} and p_t compute Δp_{t+1}

$$\| I_0 - g(I_{t+1}, p_{t+1}) \|^2 \implies \min$$

Tracking using Textured Regions

- Mean intensity difference between **I** and affine warp of template image [Shi & Tomasi, 1994]



Template I_R

Tracked state I_c

$\mathbf{I}_R$ $\mathbf{I}_C$

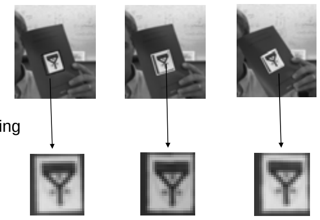
$$\psi_{tregion}(x,y) = \sum_{(x,y)\in W} (\mathbf{I}_R(x,y) - \mathbf{I}_C(x,y))^2$$

$|\mathbf{I}_R - \mathbf{I}_C|$

Template tracking: Planar Case

Planar Object => Affine motion model: $u'_i = A u_i + d$

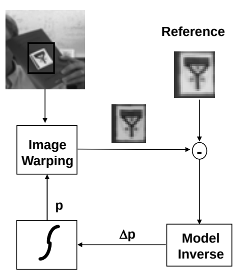
Warping



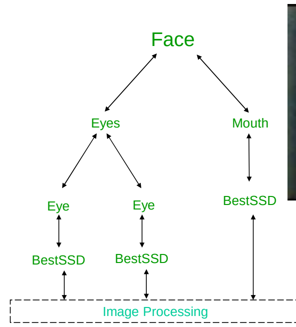
$I_t = g(p_t, I_0)$

Hager/Toyama: Tracking Cycle

- Prediction
  - Prior states predict new appearance
- Image warping
  - Generate a “normalized view”
- Model inverse
  - Compute error from nominal
- State integration
  - Apply correction to state



XVision: A tracking System



Composition of Primitive Trackers

Next Lectures

- Recognition, detection, and classification
- Reading:
  - Chapter 15: Learning to Classify
  - Chapter 16: Classifying Images
  - Chapter 17: Detecting Objects in Images