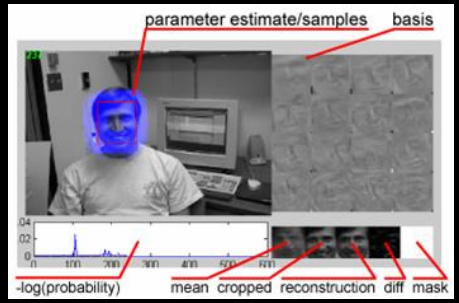
CS4495/6495
Introduction to Computer Vision

8B-L3 *Appearance-based tracking*



Visual Tracking

Conventional approaches

- Build a model before tracking starts

  Use contours, color, or appearance to represent an object

Visual Tracking

Conventional approaches

- Then do one of:

  Optical flow

  Patch tracking with particles

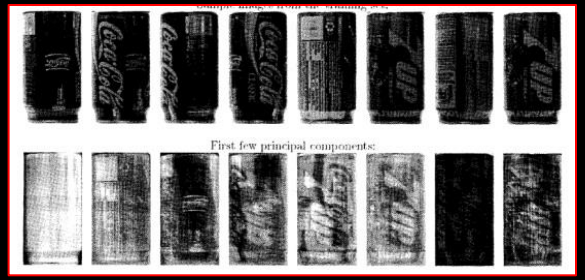
  Solve complicated optimization problem

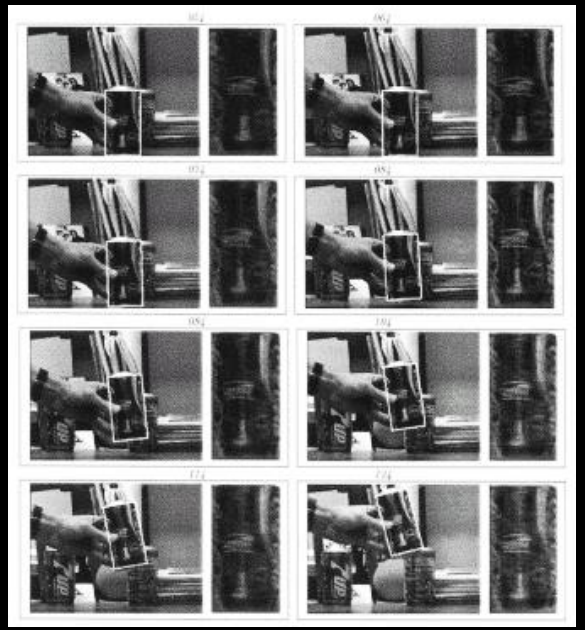
- Incorporate invariance to cope with variation in pose, lighting, view angle …

- Problem:
  - Object appearance and environments are always changing

A few key inspirations:

- Eigentracking [Black and Jepson 96]
  - View-Based learning method
  - Learn to track a “thing” rather than some “stuff”

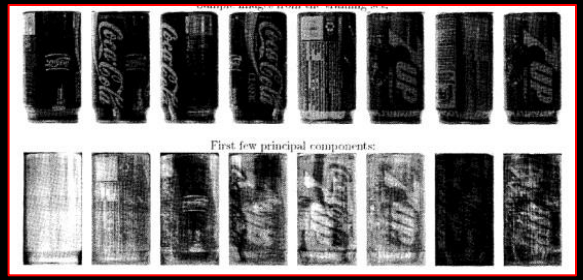


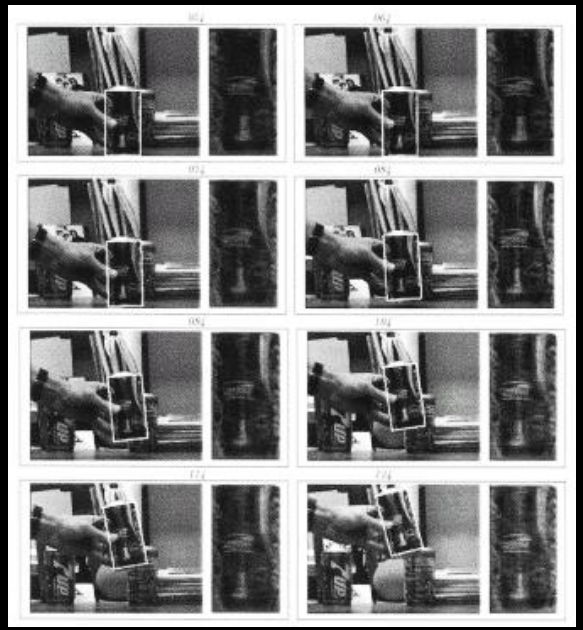


Slides from Ming-Hsuan Yang

A few key inspirations:

- Eigentracking [Black and Jepson 96]
  - Key insight: separate geometry from appearance – use deformable model
  - But…need to solve nonlinear optimization problem
  - And fixed basis set

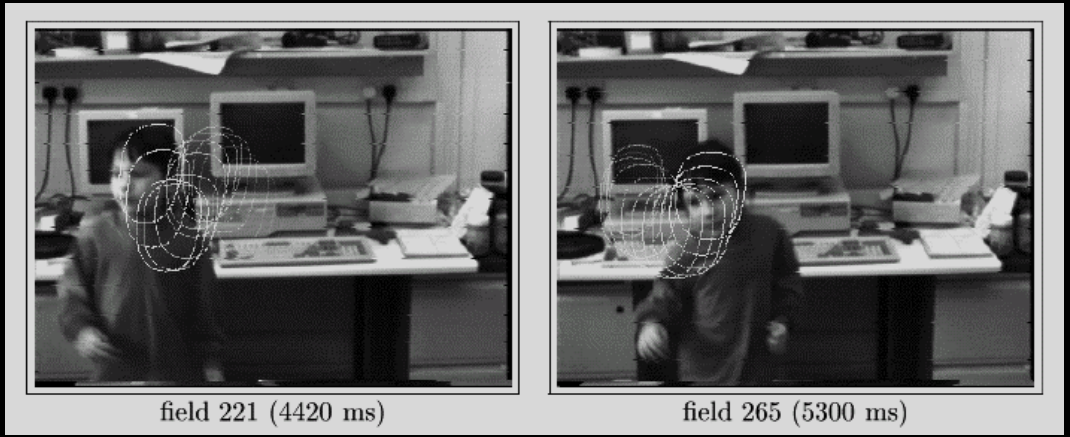




Slides from Ming-Hsuan Yang

A few key inspirations:

- Active contour [Isard and Blake 96]
  - Use particle filter
  - Propagate uncertainly over time



Incremental Visual Learning

- Aim to build a tracker that:
  - Is not single image based
  - Constantly updates the model

    Learns a representation while tracking
  - Runs fast
  - Operates on moving camera

Incremental Visual Learning

- Challenge
  - Pose variation
  - Partial occlusion
  - Illumination change
  - Drifts

Incremental Visual Learning

Int J Comput Vis (2008) 77: 125–141
DOI 10.1007/s11263-007-0075-7

Incremental Learning for Robust Visual Tracking

David A. Ross · Jongwoo Lim · Ruei-Sung Lin · Ming-Hsuan Yang

Received: 6 September 2005 / Accepted: 17 July 2007 / Published online: 17 August 2007

Abstract Visual tracking, in essence, deals with non-stationary image streams that change over time. While most existing algorithms are able to track objects well in controlled environments, they usually fail in the presence of significant variation of the object's appearance or surrounding illumination. One reason for such failures is that many algorithms employ fixed appearance models of the target. Such models are trained using only appearance data available before tracking begins, which in practice limits the range of appearances that are modeled, and ignores the large volume of information (pixel data) that becomes available during tracking. In this paper, we present a tracking method that incrementally learns a low-dimensional subspace representation, efficiently adapting online to changes in the appearance of the target. The model update, based on incremental algorithms for principal component analysis, includes two important features: a method for correctly updating the sample mean, and a forgetting factor to ensure less modeling power is expended fitting older observations. Both of these features contribute measurably to improving overall tracking performance. Numerous experiments demonstrate the effectiveness of the

Main Idea

1. Adaptive visual tracker:
   - Particle filter – draw samples from distributions of *deformation*

Slides from Ming-Hsuan Yang

Main Idea

2. Subspace-based tracking
   - Learn to track the “thing” – like the face in eigenfaces
   - Use it to determine the most likely sample

Slides from Ming-Hsuan Yang

Main Idea

3. Incremental subspace update
   - Does not need to build the model prior to tracking
   - Handles variations in lighting, pose (and expression)

Slides from Ming-Hsuan Yang

For sampling-based method

- Nomenclature
  - Location at time t: L_t
  - Current observation: F_t
- Goal: Predict target location L_t based on L_{t-1} (location in last frame)

$$p(L_t \mid F_t, L_{t-1}) \propto p(F_t \mid L_t) \cdot p(L_t \mid L_{t-1})$$

For sampling-based method

$$p(L_t \mid F_t, L_{t-1}) \propto p(F_t \mid L_t) \cdot p(L_t \mid L_{t-1})$$

- $p(L_t \mid L_{t-1}) \rightarrow$ dynamics model
  - Here use *Brownian* motion to model the dynamics – no velocity model

- $p(F_t \mid L_t) \rightarrow$ observation model
  - Use eigenbasis with approximation

State Model L_t

Representation of L_t

- *Similarity* transform: Position (x_t, y_t), rotation θ_t, scaling s_t
- 4 parameters

Or,

- *Affine* transform with 6 parameters

Dynamics Model: $p(L_t|L_{t-1})$

Simple dynamics model:

- Each parameter is independently Gaussian distributed

$$p(L_1 \mid L_0) = N(x_1; x_0, \sigma_x^2) N(y_1; y_0, \sigma_y^2) N(\theta_1; \theta_0, \sigma_\theta^2) N(s_1; s_0, \sigma_s^2)$$

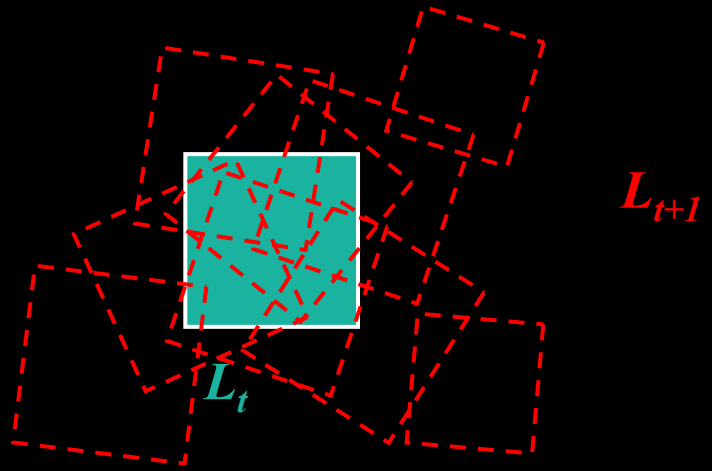
L_t

Dynamics Model: $p(L_t|L_{t-1})$

Simple dynamics model:

- Each parameter is independently Gaussian distributed

$$p(L_1 \mid L_0) = N(x_1; x_0, \sigma_x^2) N(y_1; y_0, \sigma_y^2) N(\theta_1; \theta_0, \sigma_\theta^2) N(s_1; s_0, \sigma_s^2)$$



Observation Model: $p(F_t|L_t)$

- Use probabilistic principal component analysis (PPCA) to model our image observation process

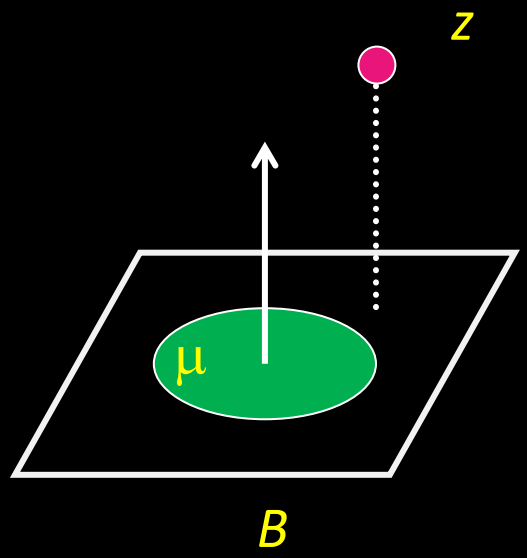
- Given a location L_t, assume the observed frame was generated from the eigenbasis

Observation Model: $p(F_t|L_t)$

The probability of observing a datum *z* given the eigenbasis *B* and mean μ,

$$p(z|B) = N(z; \mu, BB^T + \varepsilon I)$$

where εI is additive Gaussian noise

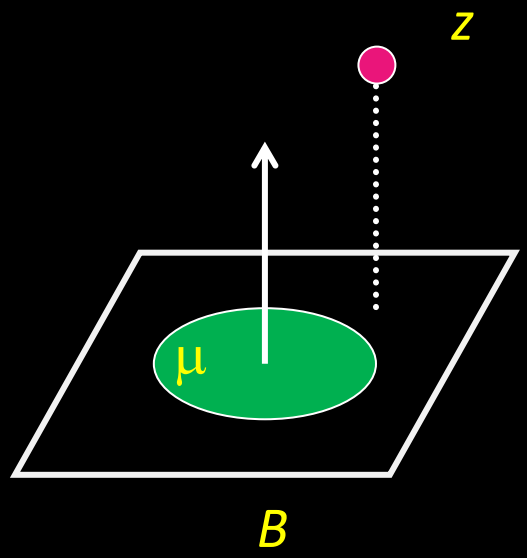


Observation Model: $p(F_t|L_t)$

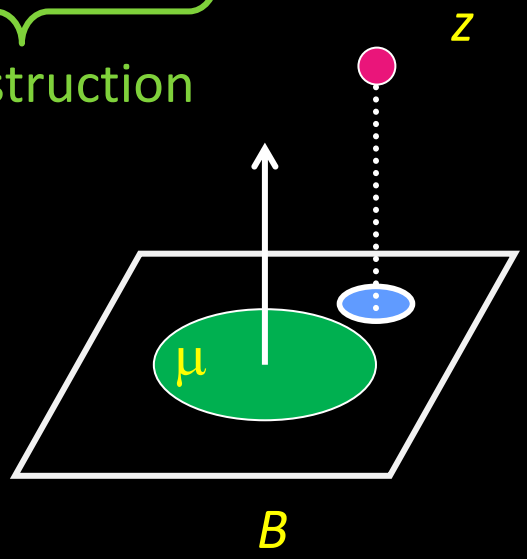
In the limit $\varepsilon \rightarrow 0$

$$p(z|B) = N(z; \mu, BB^T + \varepsilon I)$$

is mostly a function of the distance between *z* and the linear subspace *B*



$$p(z|B)\ \alpha\ \exp(-\|(z-\mu)-\underbrace{BB^T(z-\mu)}_{\text{reconstruction}}\|\ )$$



Why is that the reconstruction?

B^T is (big) d *x* (small) k

$B^T(z-\mu)$ is the coefficient vector (say γ) of $(z-\mu)$

Then $B\gamma$ is the reconstruction.

Inference

Fully Bayesian inference needs to compute

$$p(L_t | F_t, F_{t-1}, F_{t-2}, \ldots, F_0, L_0)$$

- Could do full particle filtering which is an approximation of full Bayesian inference

"But still a lot of work" (uh, maybe…)

Inference

Instead, approximate with $p(L_t \mid F_t, l^*_{t-1})$

where l^*_{t-1} is the best prediction at time *t-1*

- Different from what we did in original particle filters – here we're keeping only the "*best*" from last time-step

Maximum a posteriori estimate

Sampling for proposal

Basic tracking algorithm – given a current location:

1. Consider a number of sample locations l_s from our prior $p(L_t|l^*_{t-1})$. Their "weight" is from the prior.
2. For each sample l_s, we compute the posterior $p_s = p(l_s|F_t, l^*_{t-1})$ by using Bayes rule
3. Choose new best location (maximum a posteriori):

$$l^*_t = \arg\max p(l_s|F_t, l^*_{t-1})$$

The cool part…

Do not assume the observation model remains fixed over time

- We allow for incremental update of our object model

Given initial eigenbasis B_{t-1} and new observation w_{t-1}, *compute a new eigenbasis* B_t

- B_t is then used in $p(F_t|L_t)$

Incremental Subspace Update

- Why? To account for appearance change due to pose, illumination, shape variation.
  - Learn an *appearance representation* while tracking
- Based on the R-SVD algorithm [Golub and Van Loan 96] and the sequential Karhunen-Loeve algorithm [Levy and Lindebaum 00]
- Essentially: Develop an update algorithm with respect to running mean, allowing for decay

Put All Together

1. (Optional) Construct an initial eigenbasis if necessary (e.g., for initial detection)
2. Choose initial location L_0
3. Generate possible locations: $p(L_t | L_{t-1})$
4. Evaluate possible locations: $p(F_t | L_{t-1})$
5. Select the most likely location by Bayes: $p(L_t | F_t, L_{t-1})$
6. Update eigenbasis using R-SVD algorithm
7. Go to step 3

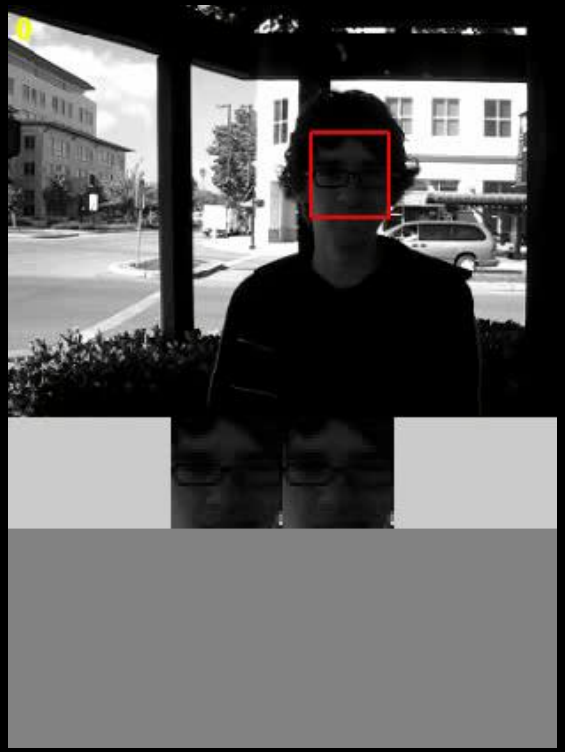
Slides from Ming-Hsuan Yang

Experiments

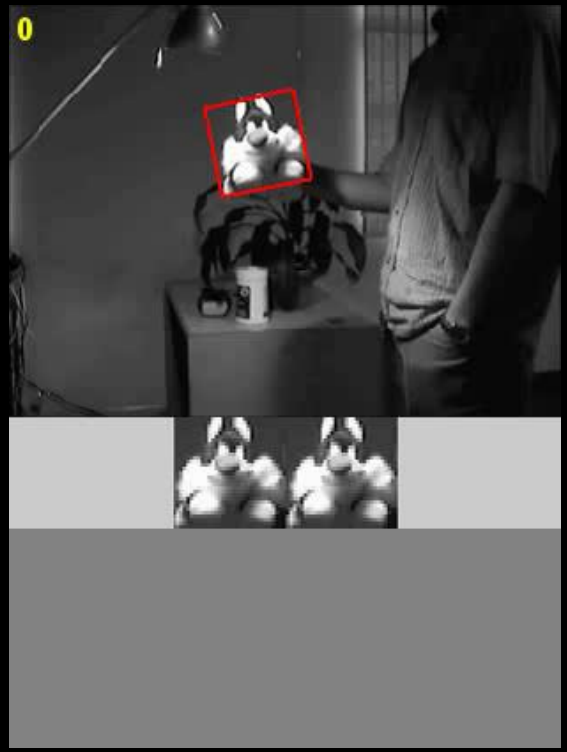
- 30 frame per second with 320 $\times$ 240 pixel resolution on old machines…
- Draw at most 500 samples
- 50 eigenvectors
- Update every 5 frames
- Runs XXX frames per second using Matlab

Slides from Ming-Hsuan Yang

Results



Large lighting variation



Plush toy tracking

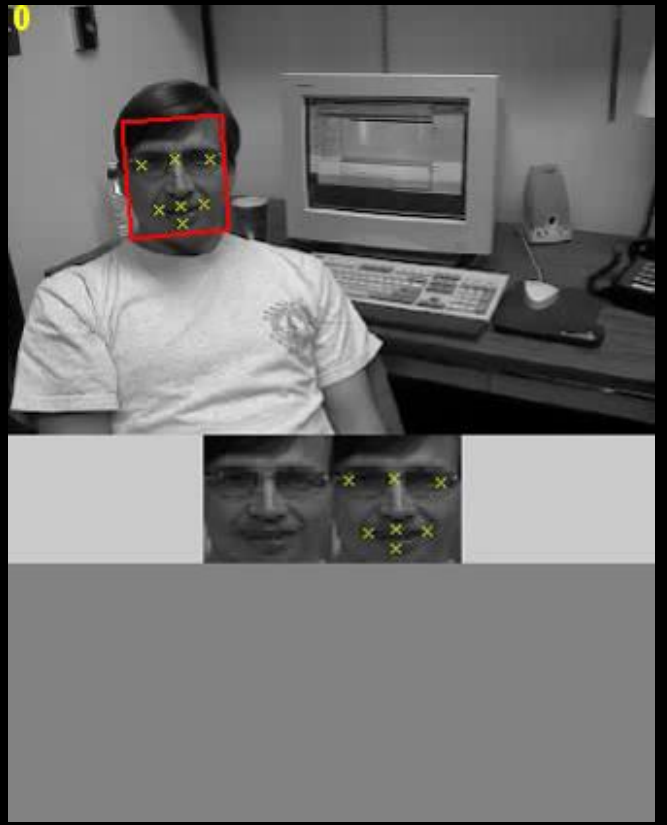
Most Recent Work

- Handling occlusion…

Occlusion Handling

An iterative method to compute a *weight mask*

- Estimate the probability a pixel is being occluded



Occlusion Handling

Given an observation I_t, initially assume there is no occlusion with W^0

$$D^{(i)} = W^{(i)} .* (I_t - UU^T I_t)$$

$$W^{(i+1)} = \exp(-D^{(i)2} / \sigma^2)$$

where .* is element-wise multiplication