Incremental Learning for Visual Tracking

Jongwoo Lim, David Ross, Ruei-Sung Lin, Ming-Hsuan Yang

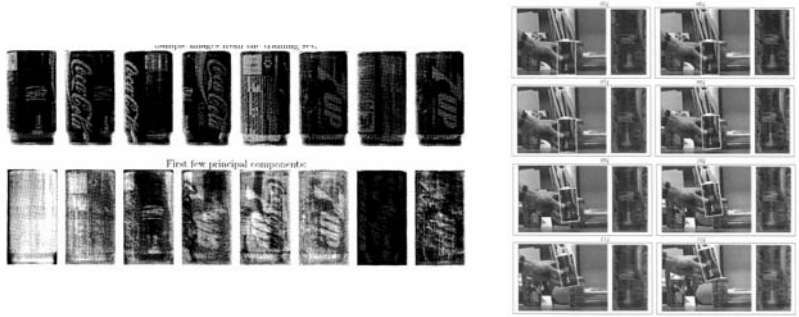
NIPS04

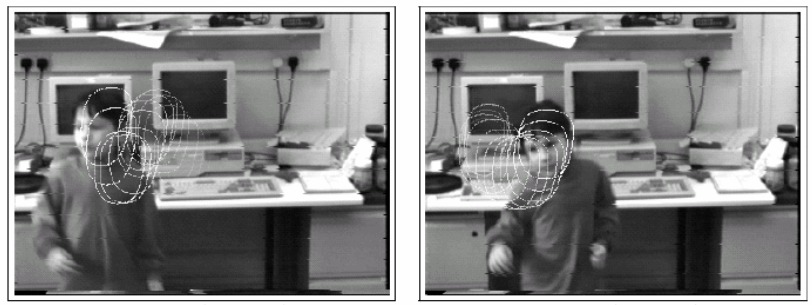
Visual Tracking

- Conventional approach
  - Build a model before tracking starts
  - Use contours, color, or appearance to represent an object
  - Optical flow
  - Incorporate invariance to cope with variation in pose, lighting, view angle …
  - View-based approach
  - Solve complicated optimization problem
- Problem:
  - Object appearance and environments are always changing

Prior Art

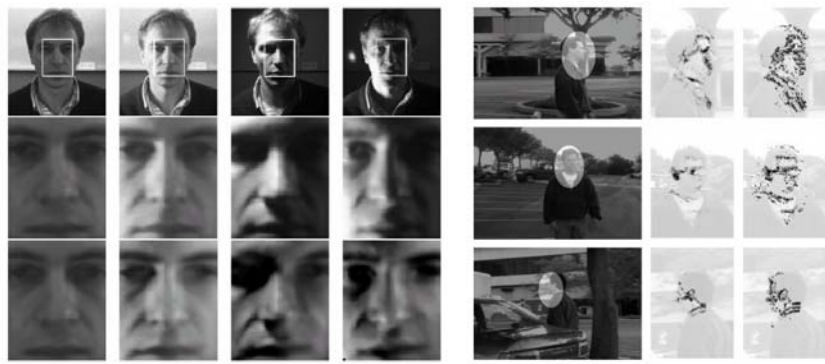
- Eigentracking [Black et al. 96]
  - View-Based learning method
  - Learn to track a "thing" rather than some "stuff"
  - Need to solve nonlinear optimization problem
- Active contour [Isard and Blake 96]
  - Use importance sampling
  - Propagate uncertainly over time
  - Edge information is sensitive to lighting change
- Gradient-Based [Shi and Tomasi 94, Hager and Belhumeur 96]
  - Gradient-Based
  - Construct illumination cone per person to handle lighting change
  - Template-based
- WSL model [Jepson et al. 2001]
  - Model each pixel as a mixture of Gaussian (MoG)
  - On-line learning of MoG
  - Track "stuff"





field 221 (4420 ms)

field 265 (5300 ms)



Incremental Visual Learning

- Aim to build a tracker that:
  - Is not view-based
  - Constantly updates the model
  - Runs fast (close to real time)
  - Tracks “thing” (structure information) rather than “stuff” (collection of pixels)
  - Operates on moving camera
  - Learns a representation while tracking
- Challenge
  - Pose variation
  - Partial occlusion
  - Adaptive to new environment
  - Illumination change
  - Drifts

Main Idea

- Adaptive visual tracker:
  - Particle filter algorithm
    - draw samples from distributions
    - does not need to solve nonlinear optimization problems
  - Subspace-based tracking
    - learn to track the "thing"
    - use it to determine the most likely sample
  - With incremental update
    - does not need to build the model prior to tracking
    - handle variation in lighting, pose and expression
- Performs well with large variation in
  - Pose
  - Lighting (cast shadows)
  - Rotation
  - Expression change
- Joint work with David Ross (Toronto) and Jongwoo Lim (UIUC/UCSD), Ruei-Sung Lin (UIUC)

Two Sampling Algorithms

- A simple sampling method with incremental subspace update [ECCV04]

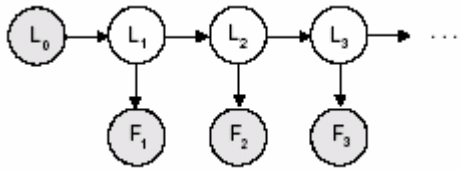
  (joint work with David Ross and Jongwoo Lim)

- A sequential inference sampling method with a novel subspace update algorithm [NIPS05a,NIPS05b]

  (joint with David Ross, Jongwoo Lim and Ruei-Sun Lin)

Graphical Model for Inference

- Given the current location L_t and current observation F_t, predict the target location L_{t+1} in the next frame

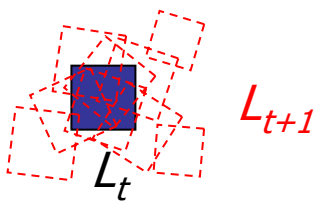


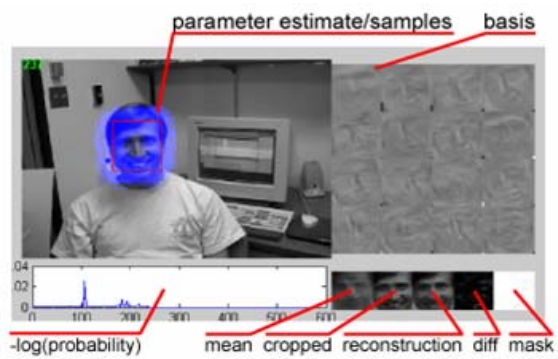
- $p(L_t|F_t, L_{t-1}) \alpha p(F_t|L_t)\, p(L_t|L_{t-1})$
- $p(L_t|L_{t-1})$ → dynamic model
  - Use Brownian motion to model the dynamics
- $p(F_t|L_t)$ → observation model
  - Use eigenbasis with approximation

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

- Representation of L_t:
  - Position (x_t,y_t), rotation (r_t), and scaling (s_t)
  - L_t=(x_t,y_t,r_t,s_t)
  - Or affine transform with 6 parameters
- Simple dynamics model:
  - Each parameter is independently Gaussian distributed

$$p(L_1|L_0) = N(x_1; x_0, \sigma_x^2)N(y_1; y_0, \sigma_y^2)N(r_1; r_0, \sigma_r^2)N(s_1; s_0, \sigma_s^2) \qquad (1)$$





Observation Model: $p(F_t | L_t)$

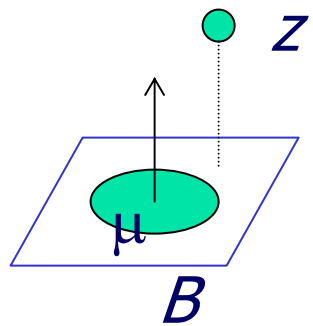
- Use probabilistic principal component analysis (PPCA) to model our image observation process
- Given a location L_t, assume the observed frame was generated from the eigenbasis
- The probability of observing a datum z given the eigenbasis B and mean μ,

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

where εI is additive Gaussian noise

- In the limit $\varepsilon \rightarrow 0$, $N(z; \mu, BB^T + \varepsilon I)$ is proportional to negative exponential of the square distance between z and the linear subspace B, i.e.,

$$p(z|B) \propto ||(z-\mu) - BB^T(z-\mu)||$$
$$p(F_t|L_t) \propto ||(F_t-\mu) - BB^T(F_t-\mu)||$$



Inference

- Fully Bayesian inference needs to compute $p(L_t | F_t, F_{t-1}, F_{t-2}, \ldots, L_0)$
- Need approximation since it is infeasible to compute in a closed form
- Approximate with $p(L_t | F_t, l^*_{t-1})$ where l^*_{t-1} is the best prediction at time t-1

Sampling Comes to the Rescue

- Draw a number of sample locations from our prior $p(L_t | l^*_{t-1})$
- For each sample l_s, we compute the posterior $p_s = p(l_s | F_t, l^*_{t-1})$
- p_s is the likelihood of l_s under our PPCA distribution, times the probability with which l_s is sampled
- Maximum a posteriori estimate

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

Remarks

- Specifically do not assume the probability of observation remains fixed over time
- To allow for incremental update of our object model
- Given an initial eigenbasis B_{t-1}, and a new observation w_{t-1}, we compute a new eigenbasis B_t
- B_t is then used in $p(F_t | L_t)$

Incremental Subspace Update

- To account for appearance change due to pose, illumination, shape variation
- Learn a representation while tracking
- Based on the R-SVD algorithm [Golub and Van Loan 96] and the sequential Karhunen-Loeve algorithm [Levy and Lindebaum 00]
- First assume zero (or fixed) mean
- Develop an update algorithm with respect to running mean

R-SVD Algorithm

- Let $X = U\Sigma V^T$ and new data Y
- Decompose Y into projection of Y onto U and its complement, $L = U^T Y, \quad H = Y - UL = (I - UU^T)Y$
- Let $Y = UL + JK$ where $JK = QR(H)$
- SVD of $[X \quad Y]$ can be written as

$$[X \quad Y] = [U \quad J] \begin{bmatrix} \Sigma & L \\ 0 & K \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^T$$

- Compute SVD of $\begin{bmatrix} \Sigma & L \\ 0 & K \end{bmatrix} = U'\Sigma'V'^T$
- Then SVD of $[X \quad Y] = U''\Sigma''V''^T$ where

$$U'' = [U \quad J]\, U' \qquad \Sigma'' = \Sigma' \qquad V'' = \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix} V'$$

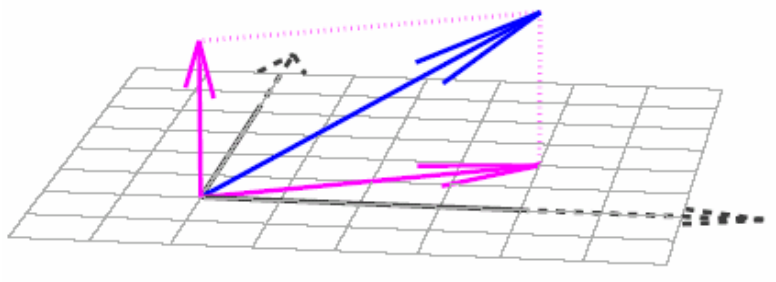
Schematically

- Decompose a vector into components within and orthogonal to a subspace

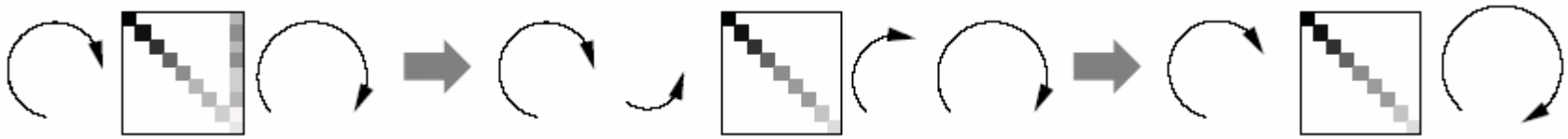
$$L = U^T Y, \quad H = Y - UL = (I - UU^T)Y$$

$$Y = UL + JK$$

$$JK = QR(H)$$



- Compute a smaller SVD



$$[X \quad Y] = [U \quad J] \begin{bmatrix} \Sigma & L \\ 0 & K \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^T = U''\Sigma''V''^T \qquad \begin{bmatrix} \Sigma & L \\ 0 & K \end{bmatrix} = U'\Sigma'V'^T$$

$$U'' = [U \quad J]\, U' \qquad \Sigma'' = \Sigma' \qquad V'' = \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix} V'$$

Put All Together

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

Experiments

- 30 frame per second with 320 $\times$ 240 pixel resolution
- Draw at most 500 samples
- 50 eigenvectors
- Update every 5 frames
- Runs 8 frames per second using Matlab
- Results
  - Large pose variation
  - Large lighting variation
  - Previously unseen object

Most Recent Work

- Subspace update with correct mean
- Sequential inference model
- Tracking without building a subspace *a priori*
- Learn a representation while tracking
- Better observation model
- Handling occlusion

Sequential Inference Model

- Let X_t be the hidden state variable describing the motion parameters. Given a set of observed images $\underline{I}_t = \{I_1, .., I_t\}$ and use Baye's theorem,

  $p(X_t \mid \underline{I}_t) \alpha\ p(I_t \mid X_t) \int p(X_t \mid X_{t-1}) p(X_{t-1} \mid \underline{I}_{t-1}) dX_{t-1}$
- Need to compute
  - Dynamic model: $p(X_t \mid X_{t-1})$
  - Observation model: $p(I_t \mid X_t)$
- Use a Gaussian distribution for dynamic model

  $p(X_t \mid X_{t-1}) = N(X_t ; X_{t-1}, \Psi)$

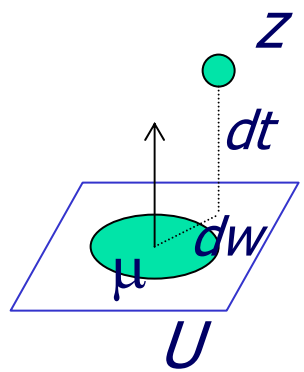
Observation Model

- Use probabilistic PCA to model the observation with
  - distance to subspace:

    $p_{dt}(I_t \mid X_t) = N(I_t;\ \mu,\ UU^T + \varepsilon I)$
  - distance within subspace:

    $p_{dw}(I_t \mid X_t) = N(I_t;\ \mu,\ U\Sigma^{-2}U^T)$
- It can be shown that

  $p(I_t \mid X_t) = p_{dt}(I_t \mid X_t)\ p_{dw}(I_t \mid X_t) =$
  $N(I_t;\ \mu,\ UU^T + \varepsilon I)\ N(I_t;\ \mu,\ U\Sigma^{-2}U^T)$



Incremental Update of Eigenbasis

- The R-SVD or SKL method assumes a fixed sample mean, i.e., uncentered PCA
- We derive a method to incremental update the eigenbasis with correct sample mean
- See [Joiffle 96] for arguments on centered and uncentered PCA

R-SVD with Updated Mean

- Proposition: Let $I_p=\{I_1, \ldots, I_n\}$, $Iq=\{I_{n+1}, \ldots, I_{n+m}\}$, and $Ir=\{I_1, \ldots, I_n, I_{n+1}, \ldots, I_{n+m}\}$. Denote the means and the scatter matrices of I_p, I_q, I_r as μ_p , μ_q , μ_r, and S_p, S_q, S_r respectively, then

$$S_r = S_p + S_q + nm/(n+m)(\mu_p - \mu_q)(\mu_p - \mu_q)^T$$

- Let $\hat{I}_p = I_p - \mu_p, \hat{I}_q = I_q - \mu_q$ and use this proposition, we get scatter matrix with correct mean

$$S_r = \begin{bmatrix} \hat{I}_p & \hat{I}_q & \sqrt{\frac{nm}{n+m}}(\mu_p - \mu_q) \end{bmatrix} \begin{bmatrix} \hat{I}_p \\ \hat{I}_q \\ \sqrt{\frac{nm}{n+m}}(\mu_p - \mu_q) \end{bmatrix}$$

- Thus, modify the R-SVD algorithm with $Y = \begin{bmatrix} I_q & \sqrt{\frac{nm}{n+m}}(\mu_p - \mu_q) \end{bmatrix}$ and the rest is the same

Occlusion Handling

- An iterative method to compute a weight mask
- Estimate the probability a pixel is being occluded
- Given an observation I_t, and initially assume there is no occlusion with $W^{(0)}$

$$D^{(i)} = W^{(i)} .* (\mathbf{I}_t - UU^T\mathbf{I}_t)$$

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

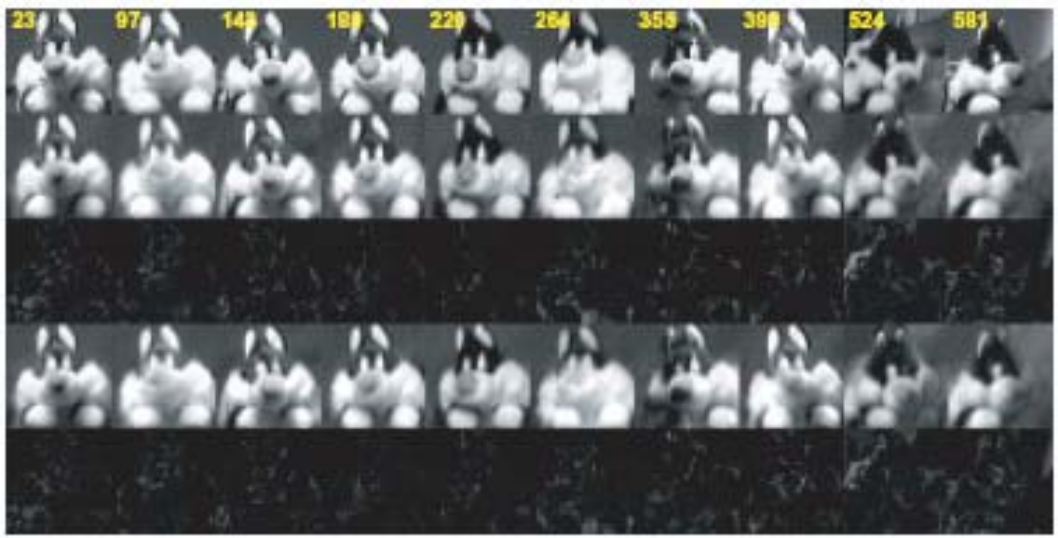
where .* is a element-wise multiplication

Experiments

- Videos are recorded with 15 or 30 frames per second with 320 x 240 gray scale images
- Use 6 affine motion parameters
- 16 eigenvectors
- Normalize images to 32 x 32 pixels
- Update every 5 frames
- Run at 4 frames per second
- Results:
  - Dudek sequence (partial occlusion)
  - Sylvester Jr (30 frame per second with cluttered background)
  - Large lighting variation with moving camera (15 frame per second)
  - Heavily shadowed condition with moving camera (15 frame per second)

Does the Increment Update Work Well?

- Compare the results
  - 121 incremental updates (every 5 frame) using our method
  - Use all 605 images with conventional PCA



tracking results

reconstruction using our method

residue: 5.65 x 10^{-2} per pixel

reconstruction using all images

residue: 5.73 x 10^{-2} per pixel

Future Work

- Verification incoming samples
- Construct global manifold
- Handling drifts
- Learning the dynamics
- Recover from failure
- Infer 3D structure from video stream
- Analyze lighting variation

Concluding Remarks

- Adaptive tracker
- Works with moving camera
- Handle variation in pose, illumination, and shape
- Handle occlusions
- Learn a representation while tracking
- Reasonable fast