Appearance-Based Tracking

current frame + previous location

Response map
(confidence map; likelihood image)

Mode-Seeking
(e.g. mean-shift; Lucas-Kanade; particle filtering)

appearance model
(e.g. image template, or
color; intensity; edge histograms)

current location
Motivation for Online Adaptation

First of all, we want succeed at persistent, long-term tracking!

The more invariant your appearance model is to variations in lighting and geometry, the less specific it is in representing a particular object. There is then a danger of getting confused with other objects or background clutter.

Online adaptation of the appearance model or the features used allows the representation to have retain good specificity at each time frame while evolving to have overall generality to large variations in object/background/lighting appearance.
Tracking as Classification

Idea first introduced by Collins and Liu, “Online Selection of Discriminative Tracking Features”, ICCV 2003

• Target tracking can be treated as a binary classification problem that discriminates foreground object from scene background.

• This point of view opens up a wide range of classification and feature selection techniques that can be adapted for use in tracking.
Overview:

Foreground samples

Background samples

Classifier

New samples

Estimated location

Response map

New frame
Observation

Tracking success/failure is highly correlated with our ability to distinguish object appearance from background.

Suggestion:

Explicitly seek features that best discriminate between object and background samples.
Continuously adapt feature used to deal with changing background, changes in object appearance, and changes in lighting conditions.

Collins and Liu, “Online Selection of Discriminative Tracking Features”, ICCV 2003
Feature Selection Prior Work

Feature Selection: choose M features from N candidates (M << N)

Traditional Feature Selection Strategies
  • Forward Selection
  • Backward Selection
  • Branch and Bound

Viola and Jones, Cascaded Feature Selection for Classification

Bottom Line: slow, off-line process
Evaluation of Feature Discriminability

Can think of this as nonlinear, “tuned” feature, generated from a linear seed feature.

Note: this example also explains why we don’t just use LDA.
Example: 1D Color Feature Spaces

Color features: integer linear combinations of R,G,B

\[
\frac{(a \, R + b \, G + c \, B)}{(|a|+|b|+|c|)} + \text{offset}
\]

where \(a, b, c\) are \{-2,-1,0,1,2\} and offset is chosen to bring result back to 0,…,255.

The 49 color feature candidates roughly uniformly sample the space of 1D marginal distributions of RGB.
Example

training frame

foreground

background

test frame

sorted variance ratio
Example: Feature Ranking
Overview of Tracking Algorithm

Note: since log likelihood images contain negative values, must use modified mean-shift algorithm as described in Collins, CVPR’03
Naive Approach to Handle Change

• One approach to handle changing appearance over time is adaptive template update

• One you find location of object in a new frame, just extract a new template, centered at that location

• What is the potential problem?
Drift is a Universal Problem!

Example courtesy of Horst Bischof. Green: online boosting tracker; yellow: drift-avoiding “semisupervised boosting” tracker (we will discuss it later today).
Template Drift

• If your estimate of template location is slightly off, you are now looking for a matching position that is similarly off center.

• Over time, this offset error builds up until the template starts to “slide” off the object.

• The problem of drift is a major issue with methods that adapt to changing object appearance.
Anchoring Avoids Drift

This is an example of a general strategy for drift avoidance that we’ll call “anchoring”.

The key idea is to make sure you don’t stray too far from your initial appearance model.

Potential drawbacks?

[answer: You cannot accommodate very LARGE changes in appearance.]
Avoiding Model Drift

Drift: background pixels mistakenly incorporated into the object model pull the model off the correct location, leading to more misclassified background pixels, and so on.

Our solution: force foreground object distribution to be a combination of current appearance and original appearance (anchor distribution)

anchor distribution = object appearance histogram from first frame
model distribution = (current distribution + anchor distribution) / 2

Note: this solves the drift problem, but limits the ability of the appearance model to adapt to large color changes
Examples: Tracking Hard-to-See Objects

Trace of selected features
Examples: Changing Illumination / Background

Trace of selected features
Examples: Minimizing Distractions

Current location

Feature scores

Top 3 weight (log likelihood) images
More Detail

top 3 weight (log likelihood) images
On-line Boosting for Tracking


Use boosting to select and maintain the best discriminative features from a pool of feature candidates.

- Haar Wavelets
- Integral Orientation Histograms
- Simplified Version of Local Binary Patterns
Adaboost learning

- Adaboost creates a single strong classifier from many weak classifiers

  - Initialize sample weights
  - For each cycle:
    - Find a classifier that performs well on the weighted sample
    - Increase weights of misclassified examples
  - Return a weighted combination of classifiers

From Daniel Vaquero, UCSB
OFF-line Boosting for Feature Selection

- Each weak classifier corresponds to a feature
- train all weak classifiers - choose best at each boosting iteration
- add one feature in each iteration

labeled training samples → weight distribution over all training samples

iterations

train each feature in the feature pool
chose the best one (lowest error) and calculate voting weight

update weight distribution

train each feature in the feature pool
chose the best one (lowest error) and calculate voting weight

update weight distribution

train each feature in the feature pool
chose the best one (lowest error) and calculate voting weight

strong classifier
Samples are patches

On-line Version...

one training sample

\[ h_{1,1}, h_{1,2}, \ldots, h_{1,M} \]

\[ h_{2,1}, h_{2,2}, \ldots, h_{2,m} \]

\[ h_{N,1}, h_{N,2}, \ldots, h_{N,m} \]

\[ \text{update } \alpha_1, \text{update } \alpha_2, \text{update } \alpha_N \]

\[ \text{current strong classifier } h_{\text{Strong}} \]

repeat for each training sample

initial importance \( \lambda = 1 \)

estimate importance \( \lambda \)

estimate importance \( \lambda \)

update \( \alpha_1 \), update \( \alpha_2 \), update \( \alpha_N \)

current strong classifier \( h_{\text{Strong}} \)
Ensemble Tracking

Avidan, “Ensemble Tracking,” PAMI 2007

Use online boosting to select and maintain a set of weak classifiers (rather than single features), weighted to form a strong classifier. Samples are pixels.

Each weak classifier is a linear hyperplane in an 11D feature space composed of R,G,B color and a histogram of gradient orientations.

Classification is performed at each pixel, resulting in a dense confidence map for mean-shift tracking.
During online updating:

• Perform mean-shift, and extract new pos/neg samples
• Remove worst performing classifier (highest error rate)
• Re-weight remaining classifiers and samples using AdaBoost
• Train a new classifier via AdaBoost and add it to the ensemble

Drift avoidance: paper suggests keeping some “prior” classifiers that can never be removed. (Anchor strategy).
Semi-supervised Boosting


Designed specifically to address the drift problem. It is another example of the Anchor Strategy.

Basic ideas:

• Combine 2 classifiers
  
  Prior (offline trained) $H^{\text{off}}$ and online trained $H^{\text{on}}$
  
  Classifier $H^{\text{off}} + H^{\text{on}}$ cannot deviate too much from $H^{\text{off}}$

• Semi-supervised learning framework
Supervised learning

Maximum margin
Can Unlabeled Data Help?

low density around decision boundary

Horst Bischof
TU Graz
Drift Avoidance

Key idea: samples from new frame are only used as unlabeled data!!!

Labeled data comes from first frame

Combined classifier

\[
\text{sign} \left( H^{\text{off}}(x) + H^{\text{on}}(x) \right)
\]
Drift Avoidance

Key idea: samples from new frame are only used as unlabeled data!!!

Labeled data comes from first frame

Combined classifier

\[ \text{sign} \left( H^{\text{off}}(x) + H^{\text{on}}(x) \right) \]
Examples

Green: online boosting
Yellow: semi-supervised
Bag of Patches Model


Key Idea: rather than try to maintain a set of features or set of classifiers, appearance of foreground and background is modeled directly by maintaining a set of sample patches.

KNN then determines the classification of new patches.
Drift Avoidance (keep patch model clean)

Given new patch samples to add to foreground and background:

• Remove ambiguous patches (that match both fg and bg)

• Trim fg and bg patches based on sorted knn distances. Remove those with small distances (redundant) as well as large distances (outliers).

• Add clean patches to existing bag of patches.

• Resample patches, with probability of survival proportional to distance of a patch from any patch in current image (tends to keep patches that are currently relevant).
Sample Results

Extension to video segmentation. See paper for the details.
Segmentation-based Tracking

This brings up a second general scheme for drift avoidance besides anchoring, which is to perform fg/bg segmentation.

In principle, it is could be a better solution, because your model is not constrained to stay near one spot, and can therefore handle arbitrarily large appearance change.

Simple examples of this strategy use motion segmentation (change detection) and data association.

Another way to avoid drift is to couple an object detector with the tracker.

Particularly for face tracking or pedestrian tracking, a detector is sometimes included in the tracking loop e.g. Yuan Li’s Cascade Particle Filter (CVPR 2007) or K.Okuma’s Boosted Particle Filter (ECCV 2004).

• If detector produces binary detections (I see three faces: here, and here, and here), use these as input to a data association algorithm.

• If detector produces a continuous response map, use that as input to a mean-shift tracker.
Summary

Tracking is still an active research topic.

Topics of particular current interest include:
• Multi-object tracking (including multiple patches on one object)
• Synergies between
  Classification and Tracking
  Segmentation and Tracking
  Detection and Tracking

All are aimed at achieving long-term persistent tracking in ever-changing environments.