Computer Vision II - Lecture 6

Tracking by Online Classification

08.05.2014

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Course Outline

- Single-Object Tracking
  - Background modeling
  - Template based tracking
  - Color based tracking
  - Contour based tracking
  - Tracking by online classification
  - Tracking-by-detection
- Bayesian Filtering
- Multi-Object Tracking
- Articulated Tracking

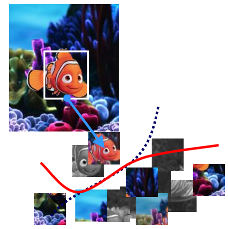


Image source: Helmut Grabner, Disney/Pixar

Recap: Deformable Contours

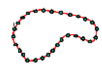
- Given
  - Initial contour (model) near desired object
- Goal
  - Evolve the contour to fit the exact object boundary

- Main ideas
  - Iteratively adjust the elastic band so as to be near image positions with high gradients, and
  - Satisfy shape "preferences" or contour priors
  - Formulation as energy minimization problem.

M. Kass, A. Witkin, D. Terzopoulos. Snakes: Active Contour Models, IJCV1988.

Recap: Energy Function

- Definition
  - Total energy (cost) of the current snake



$$E_{total} = E_{internal} + E_{external}$$

- Internal energy
  - Encourage prior shape preferences: e.g., smoothness, elasticity, particular known shape.
- External energy
  - Encourage contour to fit on places where image structures exist, e.g., edges.

$\Rightarrow$ Good fit between current deformable contour and target shape in the image will yield a low value for this cost function.

Recap: Energy Formulation

- Total energy

$$E_{total} = E_{internal} + \gamma E_{external}$$

  - with the component terms

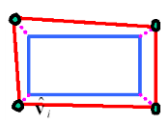
$$E_{external} = -\sum_{i=0}^{n-1} |G_x(x_i, y_i)|^2 + |G_y(x_i, y_i)|^2$$

$$E_{internal} = \sum_{i=0}^{n-1} \alpha \left(\bar{d} - \|v_{i+1} - v_i\|\right)^2 + \beta \|v_{i+1} - 2v_i + v_{i-1}\|^2$$

Behavior can be controlled by adapting the weights α, β, γ.

Recap: Extension with Shape Priors

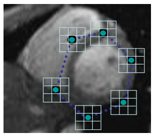
- Shape priors
  - If object is some smooth variation on a known shape, we can use a term that will penalize deviation from that shape:



$$E_{internal} += \alpha \cdot \sum_{i=0}^{n-1} (v_i - \hat{v}_i)^2$$

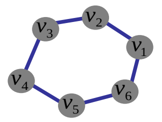
where $\{\hat{v}_i\}$ are the points of the known shape.

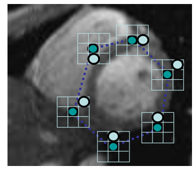
Recap: Greedy Energy Minimization



- Greedy optimization
  - For each point, search window around it and move to where energy function is minimal.
  - Typical window size, e.g., 5×5 pixels
- Stopping criterion
  - Stop when predefined number of points have not changed in last iteration, or after max number of iterations.
- Note:
  - Local optimization - need decent initialization!
  - Convergence not guaranteed

Recap: Energy Min. by Dynamic Programming



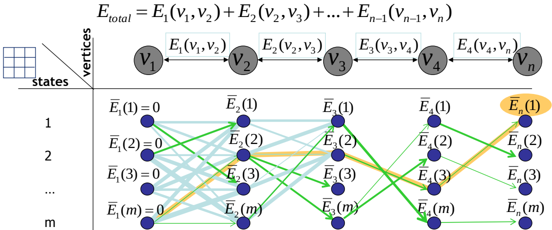


- Dynamic Programming solution
  - Limit possible moves to neighboring pixels (discrete states).
  - Find the best joint move of all points using Viterbi algorithm.
  - Iterate until optimal position for each point is the center of the box, *i.e.*, the snake is optimal in the local search space constrained by boxes.

Recap: Viterbi Algorithm

- Main idea:
  - Determine optimal state of predecessor, for each possible state
  - Then backtrack from best state for last vertex

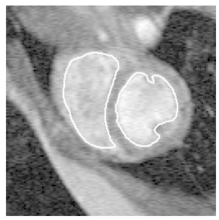
$$E_{total} = E_1(v_1,v_2) + E_2(v_2,v_3) + \ldots + E_{n-1}(v_{n-1},v_n)$$



Complexity: $\mathcal{O}(nm^2)$ vs. brute force search _____?

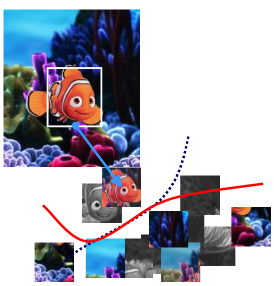
Recap: Tracking via Deformable Contours

- Idea
  1. Use final contour/model extracted at frame t as an initial solution for frame $t+1$
  2. Evolve initial contour to fit exact object boundary at frame $t+1$
  3. Repeat, initializing with most recent frame.



Tracking Heart Ventricles (multiple frames)

Today: Tracking by Online Classification



Topics of This Lecture

- Tracking by Online Classification
  - Motivation
- Recap: Boosting for Detection
  - AdaBoost
  - Viola-Jones Detector
- Extension to Online Classification
  - Online Boosting
  - Online Feature Selection
  - Results
- Extensions
  - Problem: Drift
  - Drift-compensation strategies

Tracking Requirements

- **Adaptivity**
  - Appearance changes (e.g. out of plane rotations)

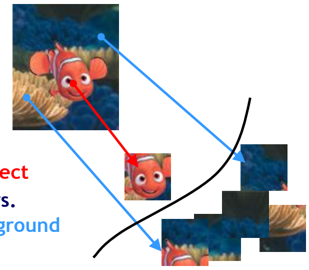
- **Robustness**
  - Occlusions, cluttered background, illumination conditions

- **Generality**
  - Any object

Tracking as Classification

- Tracking as binary classification problem

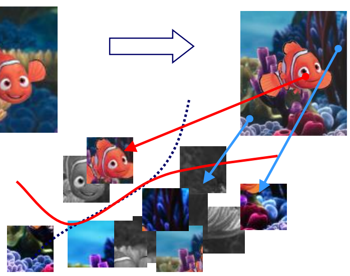
object
vs.
background



Tracking as Classification

- Tracking as binary classification problem

object
vs.
background



  - Handle object and background changes by online updating

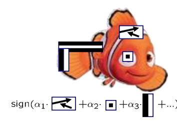
Idea: Use Boosting for Feature Selection

Object Detector

Fixed training set
General object detector

Object Tracker

On-line update
Object vs. Background



Boosting for Feature Selection

P. Viola, M. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. CVPR'01.

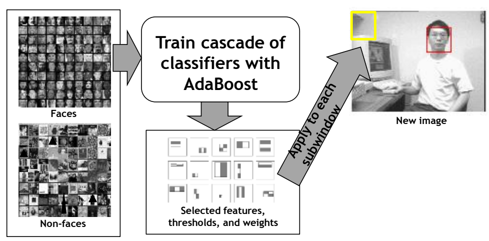
On-Line Boosting for Feature Selection

H. Grabner, H. Bischof. On-line Boosting and Vision. CVPR'06.

Topics of This Lecture

- Tracking by Online Classification
  - Motivation
- **Recap: Boosting for Detection**
  - AdaBoost
  - Viola-Jones Detector
- Extension to Online Classification
  - Online Boosting
  - Online Feature Selection
  - Results
- Extensions
  - Problem: Drift
  - Drift-compensation strategies

Recap: Viola-Jones Face Detector



- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- [Implementation available in OpenCV: http://sourceforge.net/projects/opencvlibrary/]

Recap: AdaBoost - "Adaptive Boosting"

- Main idea [Freund & Schapire, 1996]
  - Iteratively select an ensemble of classifiers
  - Reweight misclassified training examples after each iteration to focus training on difficult cases.
- Components
  - $h_m(\mathbf{x})$: "weak" or base classifier
    - Condition: <50% training error over any distribution
  - $H(\mathbf{x})$: "strong" or final classifier
- AdaBoost:
  - Construct a strong classifier as a thresholded linear combination of the weighted weak classifiers:

$$H(\mathbf{x}) = sign\left(\sum_{m=1}^{M} \alpha_m h_m(\mathbf{x})\right)$$

Recap: AdaBoost - Algorithm

1. Initialization: Set $w_n^{(1)} = \frac{1}{N}$ for $n = 1,\ldots,N$.
2. For $m = 1,\ldots,M$ iterations
   a) Train a new weak classifier $h_m(\mathbf{x})$ using the current weighting coefficients $\mathbf{W}^{(m)}$ by minimizing the weighted error function

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I(h_m(\mathbf{x}) \neq t_n) \qquad I(A) = \begin{cases} 1, & \text{if } A \text{ is true} \\ 0, & \text{else} \end{cases}$$

   b) Estimate the weighted error of this classifier on $\mathbf{X}$:

$$\epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(h_m(\mathbf{x}) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}}$$

   c) Calculate a weighting coefficient for $h_m(\mathbf{x})$:

$$\alpha_m = \ln\left\{\frac{1-\epsilon_m}{\epsilon_m}\right\}$$

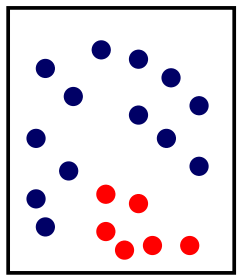
   d) Update the weighting coefficients:

$$w_n^{(m+1)} = w_n^{(m)} \exp\left\{\alpha_m I(h_m(\mathbf{x}_n) \neq t_n)\right\}$$

Topics of This Lecture

- Tracking by Online Classification
  - Motivation
- Recap: Boosting for Detection
  - AdaBoost
  - Viola-Jones Detector
- Extension to Online Classification
  - Online Boosting
  - Online Feature Selection
  - Results
- Extensions
  - Problem: Drift
  - Drift-compensation strategies

Offline Boosting



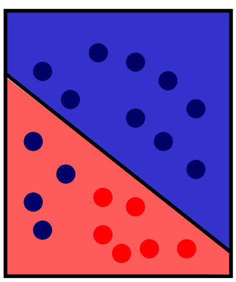
```
Given:
  - set of labeled training samples
  - weight distribution over them

Algorithm:
for n = 1 to N
  - train a weak classifier using
    samples and weight dist.
  - calculate error
  - calculate weight
  - update weight dist.
next
```

Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 1997.

Slide credit: Helmut Grabner

Offline Boosting

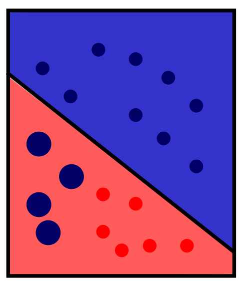


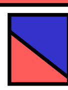
```
Given:
  - set of labeled training samples
  - weight distribution over them

Algorithm:
for n = 1 to N
  - train a weak classifier using
    samples and weight dist.
  - calculate error
  - calculate weight
  - update weight dist.
next
```

Slide credit: Helmut Grabner

Offline Boosting



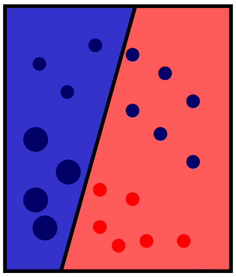
$\alpha_1 \cdot$ 

```
Given:
  - set of labeled training samples
  - weight distribution over them

Algorithm:
for n = 1 to N
  - train a weak classifier using
    samples and weight dist.
  - calculate error
  - calculate weight
  - update weight dist.
next
```

Slide credit: Helmut Grabner

Offline Boosting

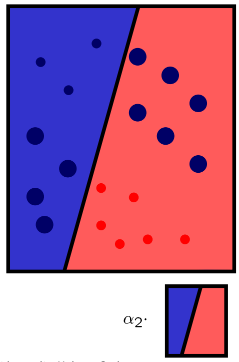


Given:
- set of labeled training samples
- weight distribution over them

Algorithm:
for n = 1 to N
- train a weak classifier using samples and weight dist.
- calculate error
- calculate weight
- update weight dist.

next

Offline Boosting



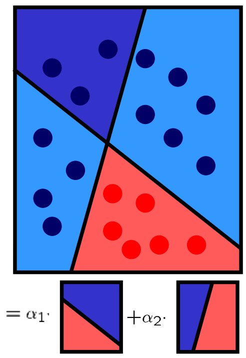
$\alpha_2 \cdot$

Given:
- set of labeled training samples
- weight distribution over them

Algorithm:
for n = 1 to N
- train a weak classifier using samples and weight dist.
- calculate error
- calculate weight
- update weight dist.

next

Offline Boosting



$= \alpha_1 \cdot$ $+ \alpha_2 \cdot$

Given:
- set of labeled training samples
- weight distribution over them

Algorithm:
for n = 1 to N
- train a weak classifier using samples and weight dist.
- calculate error
- calculate weight
- update weight dist.

next

Result:

$$h^{strong}(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}) \right)$$

From Offline to Online Boosting

- Goal
  - Formulate the algorithm such that we can present only 1 training sample at a time (and then forget about it).
  - ⇒ Dual problem: instead of keeping all samples and adding weak classifiers, keep a fixed set of weak classifiers and add samples.
- What changes?
  - Updating the classifiers online can be done easily.
    - Many classification approaches can use online updates.
  - Computing the classifier weights is also straightforward if we know the estimated error (which we can compute).

From Offline to Online Boosting

- Main issue
  - Computing the weight distribution for the samples.
  - We do not know a priori the difficulty of a sample! (Could already have seen the same sample before...)
- Idea of Online Boosting
  - Estimate the importance of a sample by propagating it through a set of weak classifiers.
  - This can be thought of as modeling the information gain w.r.t. the first n classifiers and code it by the importance weight λ for the n+1 classifier.
  - Proven [Oza]: Given the same training set, Online Boosting converges to the same weak classifiers as Offline Boosting in the limit of $N \rightarrow \infty$ iterations.

    N. Oza and S. Russell. Online Bagging and Boosting. Artificial Intelligence and Statistics, 2001.

From Offline to Online Boosting

off-line	on-line

Given:
- set of labeled training samples
  $\mathcal{X} = \{\langle \mathbf{x}_1, y_1 \rangle, ..., \langle \mathbf{x}_L, y_L \rangle \mid y_i \pm 1\}$
- weight distribution over them
  $D_0 = 1/L$

for n = 1 to N
- train a weak classifier using samples and weight dist.
  $h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$
- calculate error e_n
- calculate weight $\alpha_n = f(e_n)$
- update weight dist. D_n

next

$$h^{strong}(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}) \right)$$

From Offline to Online Boosting

RWTH AACHEN UNIVERSITY

Computer Vision II, Summer'14

off-line

Given:
- set of labeled training samples
$\mathcal{X} = \{\langle \mathbf{x}_1, y_1\rangle, ..., \langle \mathbf{x}_L, y_L\rangle \mid y_i \pm 1\}$
- weight distribution over them
$D_0 = 1/L$

for n = 1 to N
- train a weak classifier using samples and weight dist.
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$
- calculate error e_n
- calculate weight $\alpha_n = f(e_n)$
- update weight dist. D_n

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

on-line

Given:

for n = 1 to N

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

From Offline to Online Boosting

RWTH AACHEN UNIVERSITY

Computer Vision II, Summer'14

Only one training example to update the classifier

off-line

Given:
- set of labeled training samples
$\mathcal{X} = \{\langle \mathbf{x}_1, y_1\rangle, ..., \langle \mathbf{x}_L, y_L\rangle \mid y_i \pm 1\}$
- weight distribution over them
$D_0 = 1/L$

for n = 1 to N
- train a weak classifier using samples and weight dist.
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$
- calculate error e_n
- calculate weight $\alpha_n = f(e_n)$
- update weight dist. D_n

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

on-line

Given:
- ONE labeled training sample
$\langle \mathbf{x}, y\rangle \mid y \pm 1$
- strong classifier to update

for n = 1 to N

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

From Offline to Online Boosting

RWTH AACHEN UNIVERSITY

Computer Vision II, Summer'14

Update importance for the current sample

off-line

Given:
- set of labeled training samples
$\mathcal{X} = \{\langle \mathbf{x}_1, y_1\rangle, ..., \langle \mathbf{x}_L, y_L\rangle \mid y_i \pm 1\}$
- weight distribution over them
$D_0 = 1/L$

for n = 1 to N
- train a weak classifier using samples and weight dist.
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$
- calculate error e_n
- calculate weight $\alpha_n = f(e_n)$
- update weight dist. D_n

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

on-line

Given:
- ONE labeled training sample
$\langle \mathbf{x}, y\rangle \mid y \pm 1$
- strong classifier to update

- initial importance $\lambda = 1$

for n = 1 to N
- update importance weight λ

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

From Offline to Online Boosting

RWTH AACHEN UNIVERSITY

Computer Vision II, Summer'14

Online update the weak classifier

off-line

Given:
- set of labeled training samples
$\mathcal{X} = \{\langle \mathbf{x}_1, y_1\rangle, ..., \langle \mathbf{x}_L, y_L\rangle \mid y_i \pm 1\}$
- weight distribution over them
$D_0 = 1/L$

for n = 1 to N
- train a weak classifier using samples and weight dist.
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$
- calculate error e_n
- calculate weight $\alpha_n = f(e_n)$
- update weight dist. D_n

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

on-line

Given:
- ONE labeled training sample
$\langle \mathbf{x}, y\rangle \mid y \pm 1$
- strong classifier to update

- initial importance $\lambda = 1$

for n = 1 to N
- update the weak classifier using samples and importance
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(h_n^{weak}, \langle x, y\rangle, \lambda)$
- update importance weight λ

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

From Offline to Online Boosting

RWTH AACHEN UNIVERSITY

Computer Vision II, Summer'14

Update errors and weights

off-line

Given:
- set of labeled training samples
$\mathcal{X} = \{\langle \mathbf{x}_1, y_1\rangle, ..., \langle \mathbf{x}_L, y_L\rangle \mid y_i \pm 1\}$
- weight distribution over them
$D_0 = 1/L$

for n = 1 to N
- train a weak classifier using samples and weight dist.
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$
- calculate error e_n
- calculate weight $\alpha_n = f(e_n)$
- update weight dist. D_n

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

on-line

Given:
- ONE labeled training sample
$\langle \mathbf{x}, y\rangle \mid y \pm 1$
- strong classifier to update

- initial importance $\lambda = 1$

for n = 1 to N
- update the weak classifier using samples and importance
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(h_n^{weak}, \langle x, y\rangle, \lambda)$
- update error estimation $\tilde{e}_n$
- update weight $\alpha_n = f(\tilde{e}_n)$
- update importance weight λ

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

From Offline to Online Boosting

RWTH AACHEN UNIVERSITY

Computer Vision II, Summer'14

off-line

Given:
- set of labeled training samples
$\mathcal{X} = \{\langle \mathbf{x}_1, y_1\rangle, ..., \langle \mathbf{x}_L, y_L\rangle \mid y_i \pm 1\}$
- weight distribution over them
$D_0 = 1/L$

for n = 1 to N
- train a weak classifier using samples and weight dist.
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$
- calculate error e_n
- calculate weight $\alpha_n = f(e_n)$
- update weight dist. D_n

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

on-line

Given:
- ONE labeled training sample
$\langle \mathbf{x}, y\rangle \mid y \pm 1$
- strong classifier to update

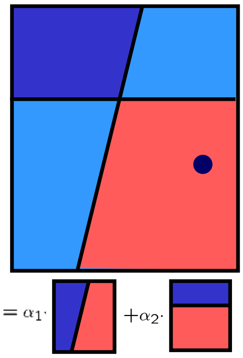
- initial importance $\lambda = 1$

for n = 1 to N
- update the weak classifier using samples and importance
$h_n^{weak}(\mathbf{x}) = \mathcal{L}(h_n^{weak}, \langle x, y\rangle, \lambda)$
- update error estimation $\tilde{e}_n$
- update weight $\alpha_n = f(\tilde{e}_n)$
- update importance weight λ

next

$h^{strong}(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x}))$

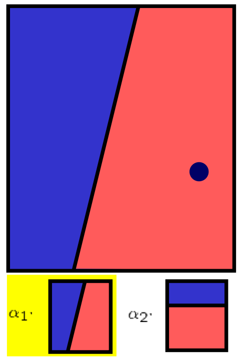
Online Boosting




```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
- initial importance
for n = 1 to N
  - update the weak classifier using
    sample and importance
  - update error estimation
  - update weight
  - update importance weight
next
```

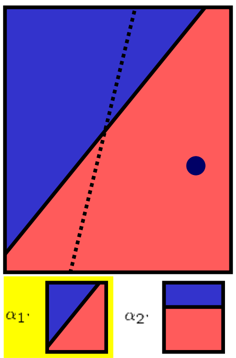

Online Boosting




```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
- initial importance
for n = 1 to N
  - update the weak classifier using
    sample and importance
  - update error estimation
  - update weight
  - update importance weight
next
```

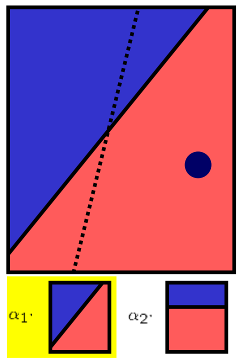

Online Boosting




```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
- initial importance
for n = 1 to N
  - update the weak classifier using
    sample and importance
  - update error estimation
  - update weight
  - update importance weight
next
```

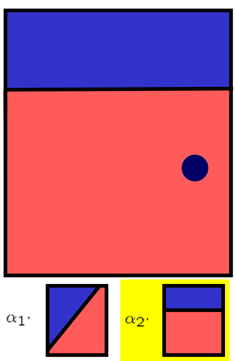

Online Boosting




```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
- initial importance
for n = 1 to N
  - update the weak classifier using
    sample and importance
  - update error estimation
  - update weight
  - update importance weight
next
```

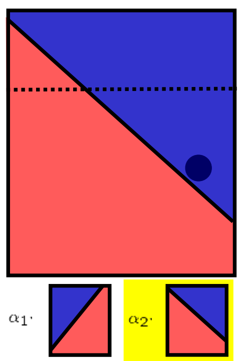

Online Boosting




```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
- initial importance
for n = 1 to N
  - update the weak classifier using
    sample and importance
  - update error estimation
  - update weight
  - update importance weight
next
```

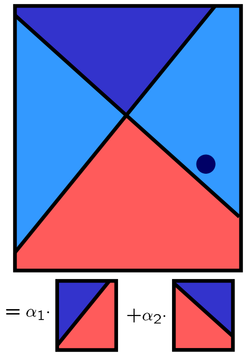

Online Boosting




```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
- initial importance
for n = 1 to N
  - update the weak classifier using
    sample and importance
  - update error estimation
  - update weight
  - update importance weight
next
```

Online Boosting




```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
- initial importance
for n = 1 to N
  - update the weak classifier using
    sample and importance
  - update error estimation
  - update weight
  - update importance weight
next

Result:
```

$$h^{strong}(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x})\right)$$

Online Boosting


```
Given:
  - ONE labeled training sample
  - strong classifier to update

Algorithm:
```

Converges to the off-line results…

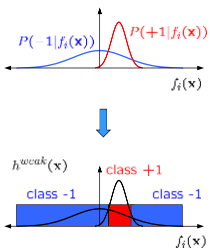
N. Oza and S. Russell. Online Bagging and Boosting. Artificial Intelligence and Statistics, 2001.

```
Result:
```

$$h^{strong}(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x})\right)$$

Online Boosting for Feature Selection

- Each feature corresponds to a weak classifier.
- Features
  - Haar-like wavelets
  - Orientation histograms
  - Locally binary patterns (LBP)
- Fast computation using efficient data structures
  - integral images
  - integral histograms



F. Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. CVPR'05.

Online Boosting for Feature Selection

- Introducing "Selector"
  - Selects one feature from its local feature pool

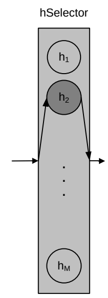
$\mathcal{H}^{weak} = \{h_1^{weak}, ..., h_M^{weak}\}$

$\mathcal{F} = \{f_1, ..., f_M\}$

$h^{sel}(\mathbf{x}) = h_m^{weak}(\mathbf{x})$

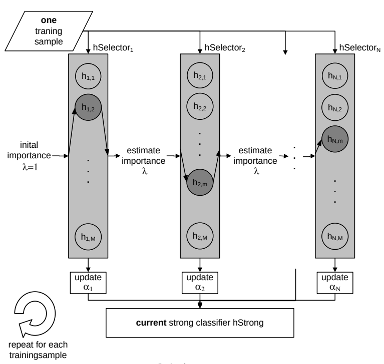
$m = \arg\min_i e_i$

On-line boosting is performed on the Selectors and not on the weak classifiers directly.

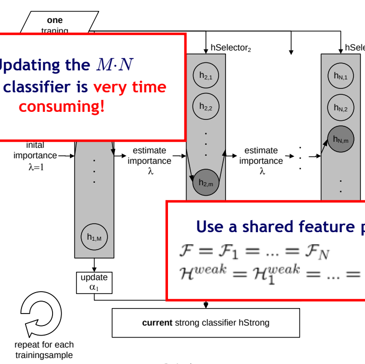


H. Grabner and H. Bischof. On-line boosting and vision. CVPR, 2006.

Online Boosting for Feature Selection



Online Boosting for Feature Selection

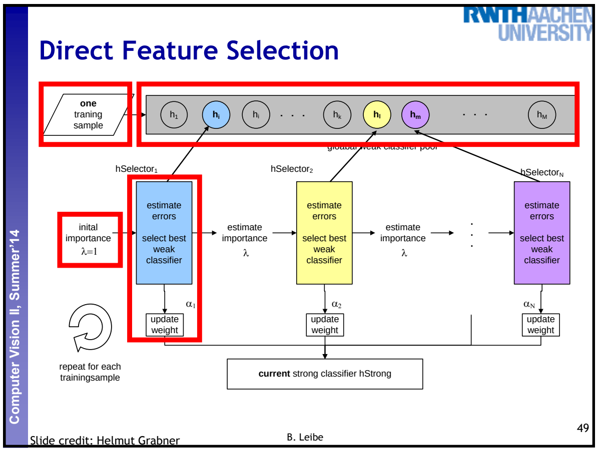


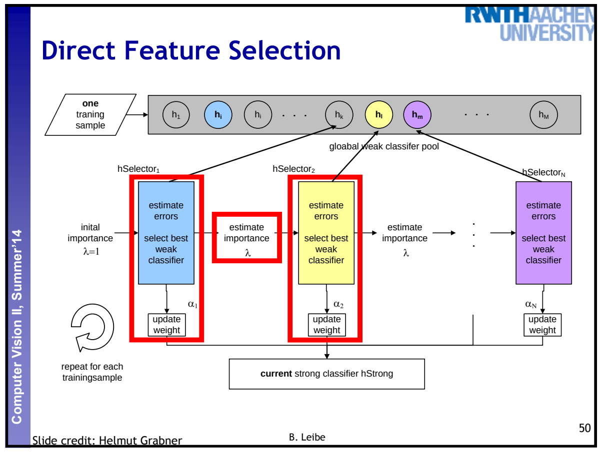
Updating the $M \cdot N$ weak classifier is very time consuming!

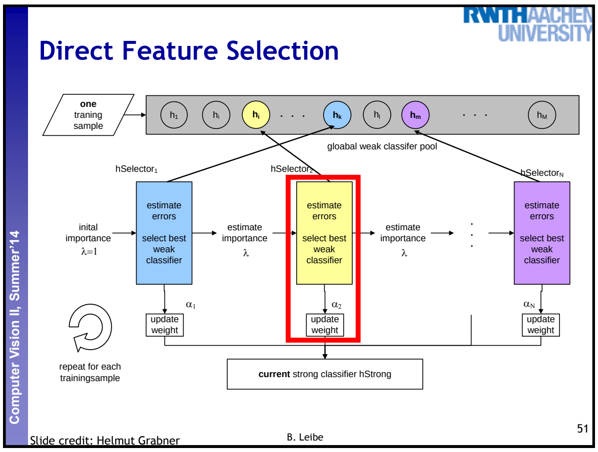
Use a shared feature pool

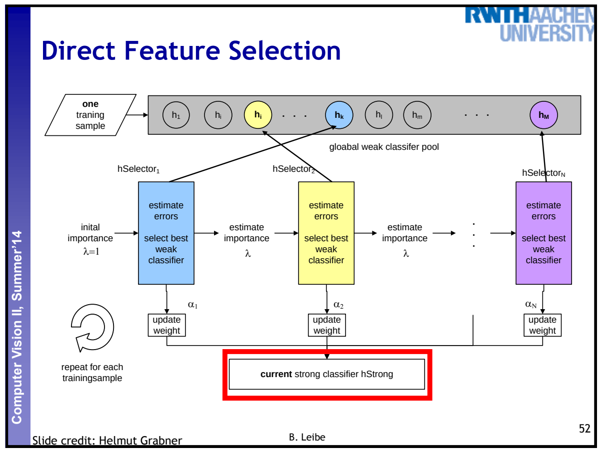
$\mathcal{F} = \mathcal{F}_1 = ... = \mathcal{F}_N$

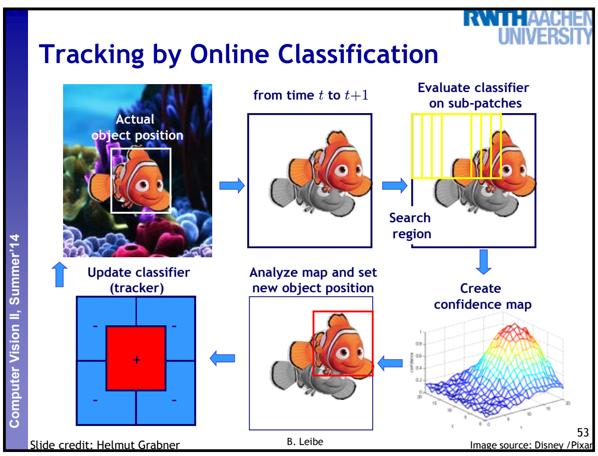
$\mathcal{H}^{weak} = \mathcal{H}_1^{weak} = ... = \mathcal{H}_N^{weak}$

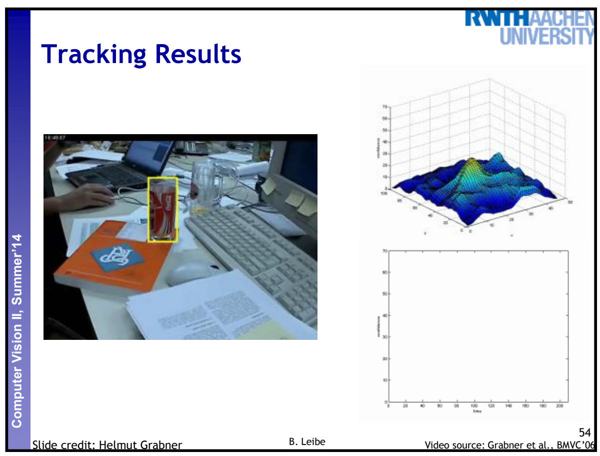

Computer Vision II, Summer'14
RWTH AACHEN UNIVERSITY
Direct Feature Selection
one traning sample
gloabal weak classifer pool
hSelector1
hSelector2
hSelectorN
inital importance λ=1
estimate errors
select best weak classifier
estimate importance λ
update weight
repeat for each trainingsample
current strong classifier hStrong
Slide credit: Helmut Grabner
B. Leibe
49


Computer Vision II, Summer'14
RWTH AACHEN UNIVERSITY
Direct Feature Selection
one traning sample
gloabal weak classifer pool
hSelector1
hSelector2
hSelectorN
inital importance λ=1
estimate errors
select best weak classifier
estimate importance λ
update weight
repeat for each trainingsample
current strong classifier hStrong
Slide credit: Helmut Grabner
B. Leibe
50

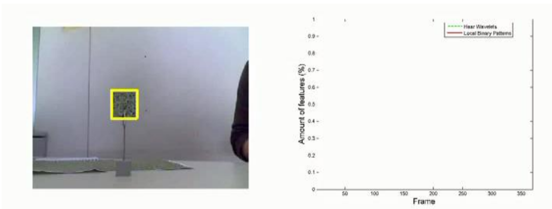

Computer Vision II, Summer'14
RWTH AACHEN UNIVERSITY
Direct Feature Selection
one traning sample
gloabal weak classifer pool
hSelector1
hSelector2
hSelectorN
inital importance λ=1
estimate errors
select best weak classifier
estimate importance λ
update weight
repeat for each trainingsample
current strong classifier hStrong
Slide credit: Helmut Grabner
B. Leibe
51


Computer Vision II, Summer'14
RWTH AACHEN UNIVERSITY
Direct Feature Selection
one traning sample
gloabal weak classifer pool
hSelector1
hSelector2
hSelectorN
inital importance λ=1
estimate errors
select best weak classifier
estimate importance λ
update weight
repeat for each trainingsample
current strong classifier hStrong
Slide credit: Helmut Grabner
B. Leibe
52


Computer Vision II, Summer'14
RWTH AACHEN UNIVERSITY
Tracking by Online Classification
Actual object position
from time t to t+1
Evaluate classifier on sub-patches
Search region
Update classifier (tracker)
Analyze map and set new object position
Create confidence map
Slide credit: Helmut Grabner
B. Leibe
53
Image source: Disney /Pixar

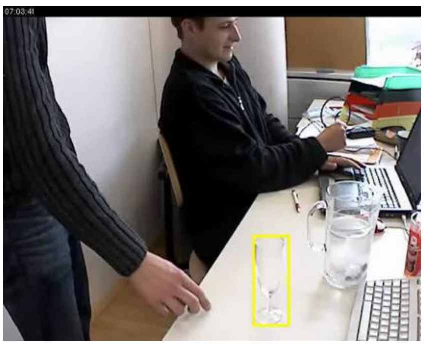

Computer Vision II, Summer'14
RWTH AACHEN UNIVERSITY
Tracking Results
Slide credit: Helmut Grabner
B. Leibe
54
Video source: Grabner et al., BMVC'06

Online Feature Exchange



Additional Tracking Results

"Tracking the Invisible"



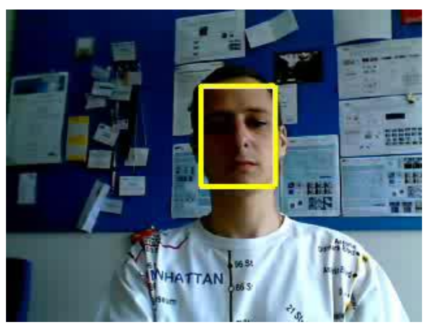
Summary: Tracking by Online Classification

- Interpret tracking as a classification problem
  - Continuously updating a classifier which discriminates the object from the background.
- Online Boosting
  - Adaptation of AdaBoost to process 1 training sample at a time.
  - Process sample by fixed set of classifiers to compute its importance weight.
  - Converges to the same result as Offline Boosting.
- Online Boosting for Feature Selection
  - Perform Boosting on Selectors instead of weak classifiers.
  - Each Selector chooses from a pool of weak classifiers.
  - Selected features and voting weights change over time.
  - Shared feature pool for real-time processing.

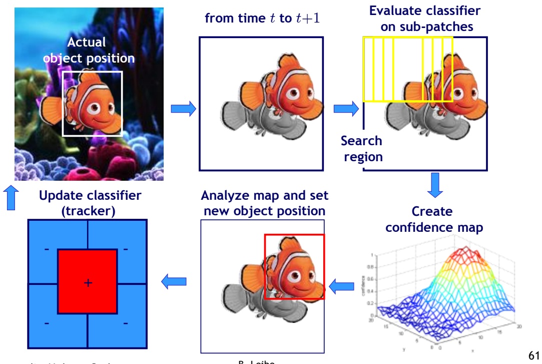
Topics of This Lecture

- Tracking by Online Classification
  - Motivation
- Recap: Boosting for Detection
  - AdaBoost
  - Viola-Jones Detector
- Extension to Online Classification
  - Online Boosting
  - Online Feature Selection
  - Results
- **Extensions**
  - **Problem: Drift**
  - **Drift-compensation strategies**

When Does It Fail...

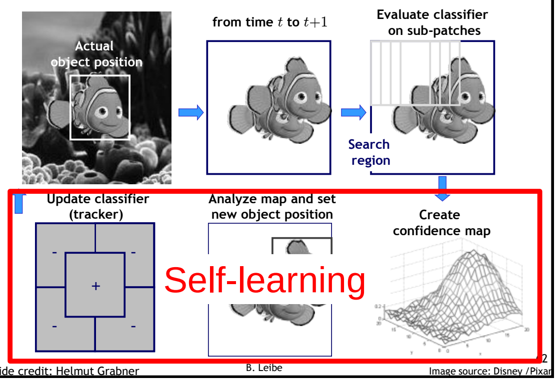


Why Does It Fail?



Slide credit: Helmut Grabner | B. Leibe | Image source: Disney /Pixar

Why Does It Fail?

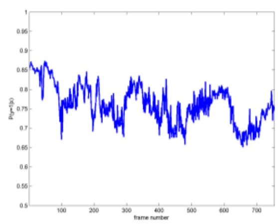


Slide credit: Helmut Grabner | B. Leibe | Image source: Disney /Pixar

Drifting Due to Self-Learning Policy

Tracked Patches

Confidence



⇒ Not only does it drift, it also remains confident about it!

Slide credit: Helmut Grabner | B. Leibe | Image source: Grabner et al., ECCV'08

Self-Learning and Drift

- Drift
  - Major problem in all adaptive or self-learning trackers.
  - Difficulty: distinguish "allowed" appearance change due to lighting or viewpoint variation from "unwanted" appearance change due to drifting.
  - Cannot be decided based on the tracker confidence!
    - Since the confidence is always dependent on the learned model
    - Model may already be affected by drift when the confidence is measured.
  - Several approaches have been proposed to address this.

B. Leibe

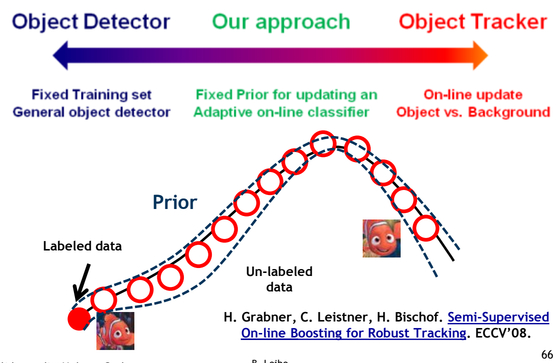
Strategy 1: Match Against Initialization

- Used mostly in low-level trackers (e.g., KLT)
  - Advantage: robustly catches drift
  - Disadvantage: cannot follow appearance changes.

J. Shi and C. Tomasi. Good Features to Track. CVPR 1994.

B. Leibe

Strategy 2: Semi-Supervised Learning



H. Grabner, C. Leistner, H. Bischof. Semi-Supervised On-line Boosting for Robust Tracking. ECCV'08.

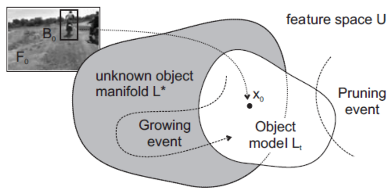
Slide credit: Helmut Grabner | B. Leibe

Tracking despite Occlusions

Object Disappearance

Long-Term Tracking (1h)

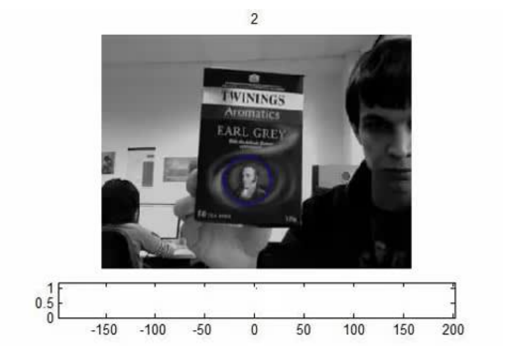
Strategy 3: Using Additional Cues



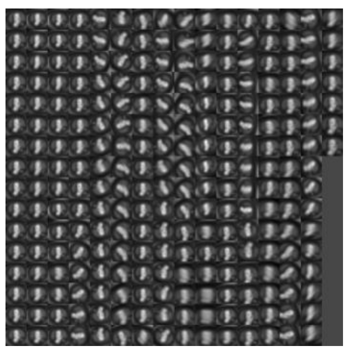
- **Tracking-Learning-Detection**
  - Combination of KLT and Tracking-by-Detection
  - Use a KLT tracker as additional cue to generate confident (positive and negative) training examples.
  - Learn an object detector on the fly using Online Random Ferns.

Z. Kalal, K. Mikolajczyk, J. Matas. Tracking-Learning-Detection. PAMI 2011.

TLD Results



Accumulated Training Examples



TLD Results

Video source: Z. Kalal

References and Further Reading

- **The original Online AdaBoost paper**
  - N. Oza and S. Russell. Online Bagging and Boosting. Artificial Intelligence and Statistics, 2001.
- **Online Boosting for Tracking**
  - H. Grabner, H. Bischof. On-line Boosting and Vision. CVPR'06.
- **Semi-Supervised Boosting**
  - H. Grabner, C. Leistner, H. Bischof. Semi-Supervised On-line Boosting for Robust Tracking. ECCV'08.
- **Tracking-Learning-Detection**
  - Z. Kalal, K. Mikolajczyk, J. Matas. Tracking-Learning-Detection. PAMI 2011.