# Multiclass object detection for robot and computer vision applications

**Rodrigo Verschae**

Advanced Mining Technology Center, AMTC
Universidad de Chile, Chile

Xerox Research Center Europe - XRCE
February 22, 2013

# Multiclass Object detection

**Object detection**: For a given set of object classes, to find, if any, the instances of these classes appearing on the image.

# What do we need from Multiclass object classifier?

## Use

- High Accuracy: detection rate $> 99.9\%$; false positive rate $< 10^{-6}$
- Fast detection & Robustness (to noise, illumination, etc.)

# What do we need from Multiclass object classifier?

## Use

- High Accuracy: detection rate $> 99.9\%$; false positive rate $< 10^{-6}$
- Fast detection & Robustness (to noise, illumination, etc.)

## Training

- Procesing time
- Good results when trained using small training sets

# What do we need from Multiclass object classifier?

## Use

- High Accuracy: detection rate $> 99.9\%$; false positive rate $< 10^{-6}$
- Fast detection & Robustness (to noise, illumination, etc.)

## Training

- Procesing time
- Good results when trained using small training sets

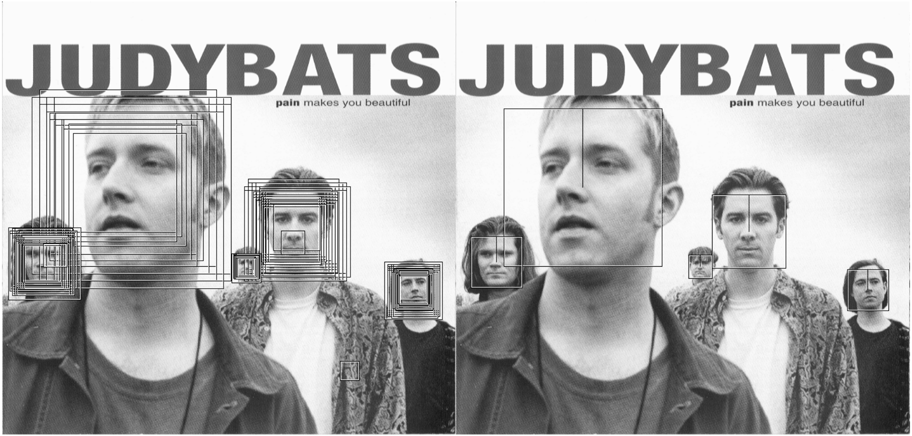
## Scalablility with the number of classes

- Detection Rate (per class)
- False positive rate
- Detection & Training time

1. Introduction
    - Cascade classifiers
    - Adaboost
2. Multiclass Object Detection
    - Multiclass boosting and weak classifiers
    - Multiclass cascade
    - TCAS: Tree of nested CAScades
3. Results, conclusions and future work

# Search: Sliding window

# Search: Sliding window



# Cascade classifiers [Viola and Jones 2001]

## Characteristic of the Problem

- Highly asymmetric problem: mostly background
- Most non-object windows are very different from object windows

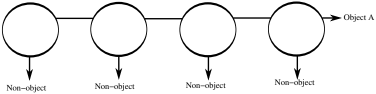
$\rightarrow$ Average processing time depends mainly on non-object windows

# Cascade classifiers [Viola and Jones 2001]

## Characteristic of the Problem

- Highly asymmetric problem: mostly background
- Most non-object windows are very different from object windows

$\rightarrow$ Average processing time depends mainly on non-object windows



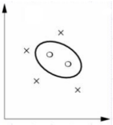

## Cascade classifier

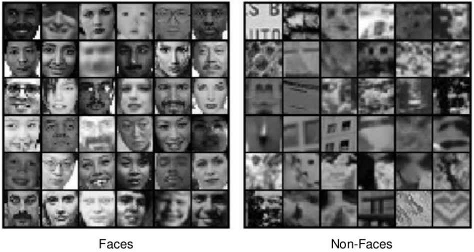
- Coarse-To-Fine approach:
  $\rightarrow$ reduce negative hypothesis space incrementally
- Efficient way of organizing object detection:
  $\rightarrow$ more non-object than object windows

# Adaboost (two-classes)

## Classification: Basic concepts

- Training set $S = \{(x_i, y_i)\}_{i=1,\dots,m}, y_i \in \{-1, 1\}$
- Build a classifier $H(x) : \mathbb{R}^n \to \mathbb{R}$
- We want $Prob(sign(H(x)) \neq y)$ to be small





# Real Adaboost (two-classes)

## Main idea

- Additive model: $H(x) = \sum_{t=0}^{T} h_t(x)$
- Add terms to sum iteratively
- Focus on wrongly classified examples
- Minimizes $E_{x \sim S}[\exp(-yH(x)]$ iteratively

Note: The classification is correct $\iff$ the margin $yH(x) \geq 0$

# Features

## Associate a weak classifier to a feature

$$H(x) = \sum_{t=0}^{T} \hat{h}_t(x) = \sum_{t=0}^{T} h_t(f_t(x))$$

# Features

## Associate a weak classifier to a feature

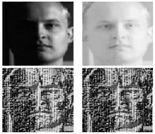
$$H(x) = \sum_{t=0}^{T} \hat{h}_t(x) = \sum_{t=0}^{T} h_t(f_t(x))$$

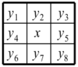
## Modified Local Binary Patterns (mLBP) [Froba et al 2004]

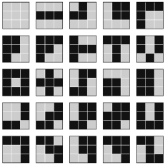
- Invariant to linear constrast changes

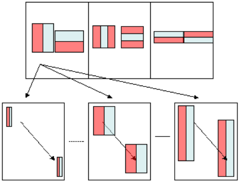
## Haar-like wavelets [Viola and Jones 2001]

- Fast evaluation using the Integral Image





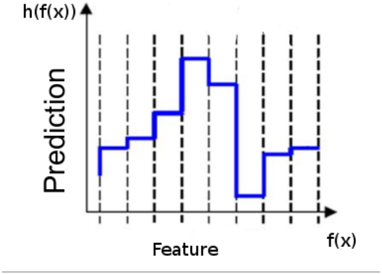





# Domain Partitioning weak classifiers

## [Shapire and Singer 1999]

- Fast evaluation thanks to the use of look up tables: $O(1)$.
- Well fitted for continue and discrete features (e.g, rectangular features and mLBP).




$$H(x) = \sum_{t=0}^{T} h_t(f_t(x))$$

# Adaboost

Input: Training set $S = \{(x_i, y_i)\}_{i=1,\ldots,m}$

- Initialize Weights: $w_i(0) = 1, i = 1, \ldots, m$
- For $t = 0, \ldots, T$
  - Normalized weigths $\{w_i(t)\}_{i=1,\ldots,m}$ such that they add to one
  - Select $h_t \in \mathbf{H}$ (and $f_t \in \mathbf{F}$) that minimizes:

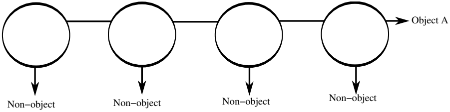
$$Z_t = \sum_{i=1}^{m} w_i(t) \exp\left(-y_i h_t(f_t(x_i))\right)$$

  - Update weigths: $w_i(t+1) = w_i(t) \exp\left(-y_i, h_t(f_t(x_i))\right)$

Output:

$$H(x) = \sum_{t=0}^{T} h_t(f_t(x))$$

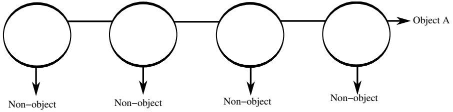
# Nested Cascade [Wu et at 2004]




## Nested cascade

$$H^k(x) = \sum_{t=0}^{T_k} h_t^k(f_t(x))$$

# Nested Cascade [Wu et at 2004]




## Nested cascade

$$H^k(x) = \sum_{t=0}^{T_k} h_t^k(f_t(x)) + H^{k-1}(x), k > 1$$

- Allows reusing information
- Faster and more robust than the non-nested case

# Training Framework

$$H^k(x) = \sum_{t=0}^{T_k} h_t^k(f_t(x)) + H^{k-1}(x) - b_k, k > 1 \quad (1)$$

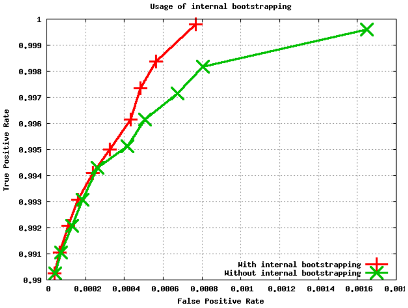
## Main Contributions [Verschae et al 2008]

- Criterion to decide when stop the training of a layer
  $\rightarrow$ trade-off: processing speed, false positives and true positives
- Selection of negative examples: Internal and external bootstrapping
- Heuristics to speed up the training (feature sampling, feature types)

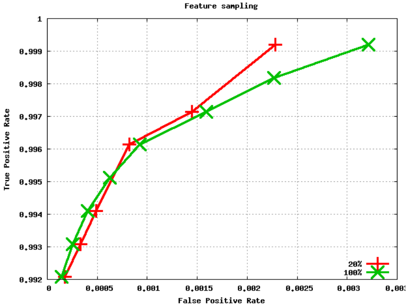
## Results

- The structure of the cascade does not have to be predefined
- Short running time ($\sim$50 ms, images of 320x240, Pentium 4 GHz)
- Short training time ($\sim$15 hours, Pentium 4 GHz, 10000 training samples per layer vs weeks in [Viola & Jones])

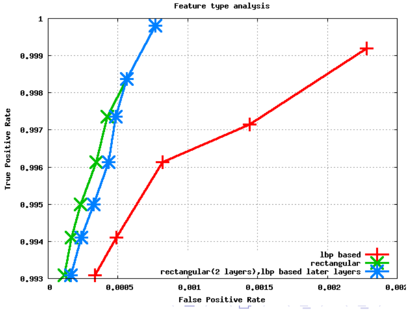
# Results



Usage of internal bootstrapping
True Positive Rate
False Positive Rate
With internal bootstrapping
Without internal bootstrapping

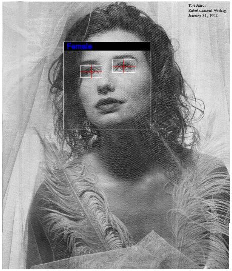



Feature sampling
True Positive Rate
False Positive Rate
20%
100%

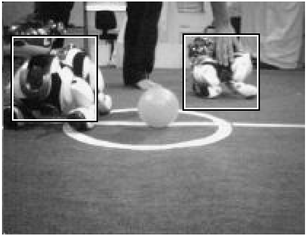



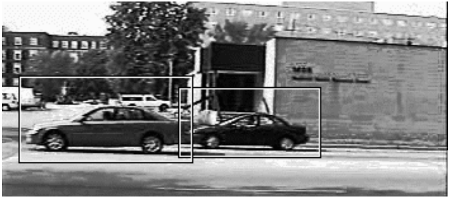
Feature type analysis
True Positive Rate
False Positive Rate
lbp based
rectangular
rectangular(2 layers),lbp based later layers

# Results: face, eye, Aibo robot, and car detection









# Multiclass Object Detection: Related work

- Multiple cascades [Wu & al 2004, Schneiderman 2004]
  $\rightarrow$ Frontal faces, rotated faces, cars, traffic signs, etc.
- View estimation followed by multiple cascades [Jones & Viola 2003]
  $\rightarrow$ Frontal and rotated faces.

# Multiclass Object Detection: Related work

- Multiple cascades [Wu & al 2004, Schneiderman 2004]
  → Frontal faces, rotated faces, cars, traffic signs, etc.
- View estimation followed by multiple cascades [Jones & Viola 2003]
  → Frontal and rotated faces.
- Tree of boosted cascades (Floatboost) [Li et al 2004 ; Ong & Bowden 2004]
  → Faces, hands.
- Coarse-To-fine [Fleuret & Geman 2001; Amit et al. 2004; Gangaputra & Geman 2006]
  → Faces, Numbers and Letters.
- Sharing of weak classifiers among classes (Jointboost) [Torralba et al. 2004]
  → frontal faces, cars, persons, keyboards, telephones, etc.
- Vectorized Adaboost and tree structure (Vector Boosting) [Huang et al 2007]
  → Faces at multiple rotations.

# Desired properties in the multiclass case

## Scalablility with the number of classes

- in accuracy, robustness, processing time
- training time, robust with small training sets.

# Desired properties in the multiclass case

## Scalablility with the number of classes

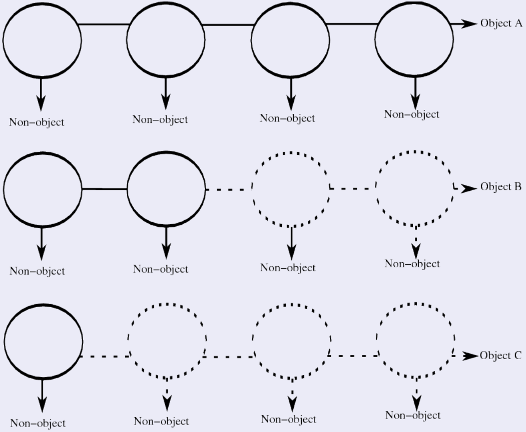
- in accuracy, robustness, processing time
- training time, robust with small training sets.

## Solution

- "Share" feature evaluations
- "Share" classifier evaluations and parameters
- Coarse-To-Fine Search:
    - in the negative space (cascade like)
    - the positive space (tree like)

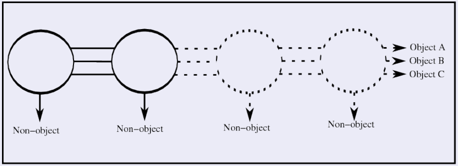
# Multiclass detector structure

## Multiple cascades



# Multiclass detector structure

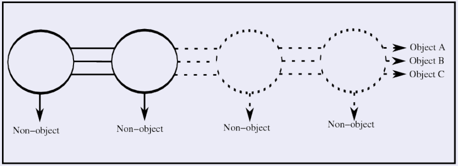
## Multiclass cascades [Verschae et al 2008]




Notation:
Classifier $\vec{H}_k(x)$ at layer $k$

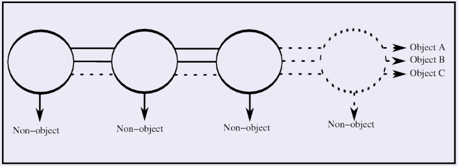
# Multiclass detector structure

## Multiclass cascades [Verschae et al 2008]



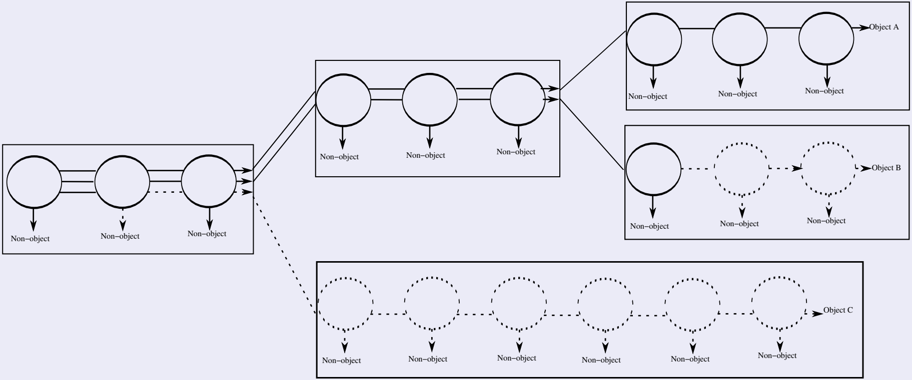

Notation:
Classifier $\vec{H}_k(x)$ at layer $k$

## Coaser-To-Fine (CTF) Multiclass cascade [Verschae et al 2010]



# Multiclass detector structure

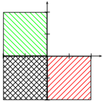
## Multiclass TCAS here proposed



# Multiclass formulation

## Multiclass boosting

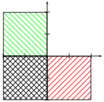
$$\vec{H}(x) = \sum_{t=0}^{T} \vec{h}_t(\vec{f}_t(x))$$



# Multiclass formulation

## Multiclass boosting

$$\vec{H}(x) = \sum_{t=0}^{T} \vec{h}_t(\vec{f}_t(x))$$



## Feature sharing

One-dimensional feature per multiclass weak classifier

$$\vec{H}(x) = \sum_{t=0}^{T} \vec{h}_t(f_t(x))$$

# Multiclass formulation

## Two-class case

- Training example: $(x_i, y_i)$
  - $y_i \in \{-1, 1\}$ represents the target halfspace
- Correct classification:
  - $yH(x) \geq 0$ if $H(x)$, i.e. is in the correct half space

# Multiclass formulation

## Two-class case

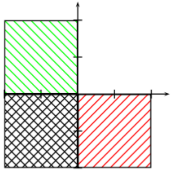
- Training example: $(x_i, y_i)$
  - $y_i \in \{-1, 1\}$ represents the target halfspace
- Correct classification:
  - $yH(x) \geq 0$ if $H(x)$, i.e. is in the correct half space

## Multiclass case

- Training example: $(x_i, \mathbf{V}_i)$, with $\mathbf{V}_i$ a vector set
  - $\mathbf{V}_i$ represents the target sub space (intersection of halfspaces)
- Correct classification:
  - iff $\forall \vec{V} \in \mathbf{V}_m, \vec{V} \cdot \vec{H}(x) \geq 0$ (Vector Boosting, [Huang et al 2007])

# Objective space

## Example 1: 2 objects classes and a non-object class

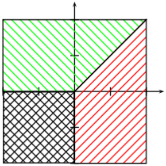


Disjoint and non overlapping

| Class | Vectors | Objective region | Region on figure |
|---|---|---|---|
| Object 1 | $(+1, 0); (0, -1)$ | $x > 0, y < 0$ | Red |
| Object 2 | $(-1, 0); (0, +1)$ | $x < 0, y > 0$ | Green |
| Non Object | $(-1, 0); (0, -1)$ | $x < 0, y < 0$ | Dashed Grey |

# Objective space

**Example 2: 2 objects classes and a non-object class**

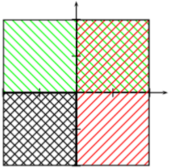


Joint boundary

| Class | Vectors | Objective region | Region on figure |
|---|---|---|---|
| Object 1 | $(+1, 0); (+1, -1)$ | $x > 0, x > y$ | Red |
| Object 2 | $(-1, 0); (-1, +1)$ | $y > 0, y > x$ | Green |
| Non Object | $(-1, 0); (0, -1)$ | $x < 0, y < 0$ | Dashed Grey |

# Objective space

Example 3: 2 objects classes and a non-object class



Ovelapping

| Class | Vectors | Objective region | Region on figure |
|---|---|---|---|
| Object 1 | $(1,0);$ | $x > 0$ | Red |
| Object 2 | $(0,1);$ | $y > 0$ | Green |
| Non Object | $(-1,0);(0,-1)$ | $x < 0, y < 0$ | Dashed Grey |

Assign to class $m$ if $\forall \vec{V} \in \mathbf{V}_m, \vec{V} \cdot \vec{H}(x) \geq 0$

# Generalized Adaboost

Training set $S = \{(x_i, \mathbf{V}_i)\}_{i=1,\dots,m}$
$\mathbf{V}_i = \{\vec{V}_i^j\}_j$: vector set representing objective region of example $x_i$

- Initialize Weights: $w_{i,j}(0) = 1$
- For $t = 0, \dots, T$
  1. Normalized weigths $\{w_{i,j}(t)\}_{i,j}$ so that they add to one
  2. Select $\vec{h}_t$ and $f_t$, such that $Z_t$ is minimized, with

$$Z_t = \sum_{i=1}^{m} \sum_{j : V_i^j \in \mathbf{V}_i} w_{i,j}(t) \exp\left(-Q\left(\vec{V}_i^j, \vec{h}_t(f_t(x_i))\right)\right)$$

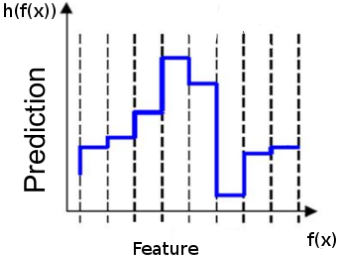
  3. Update weigths: $w_{i,j}(t+1) = w_{i,j}(t) \exp\left(-Q\left(\vec{V}_i^j, \vec{h}_t(f_t(x_i))\right)\right)$

return $\vec{H}(x) = \sum_t \vec{h}_t(f_t(x))$

# Domain Partitioning weak classifiers

## one-class case [Shapire and Singer 1999]

- Fast evaluation thanks to the use of look up tables: $O(1)$.
- Well fitted for continue and discrete features (e.g, rectangular features and mLBP).




$$H(x) = \sum_{t=0}^{T} h_t(f_t(x))$$

# Posible outputs of the weak classifier $\vec{h}_k$

- Independent classifiers [Huang et al 2007]

$$H(x,c) = \sum_{t=1}^{T} h_t(f_t(x), c)$$

- Joint classifiers Similar to [Torralba et al 2007]

$$H(x,c) = \sum_{t=1}^{T} \beta_t^c h_t(f_t(x)), \beta_t^c \in \{0,1\}$$

- Coupled classifiers [Verschae et al 2008]

$$H(x,c) = \sum_{t=1}^{T} \gamma_t^c h_t(f_t(x)), \gamma_t^c \in \mathbb{R}$$

# Posible outputs of the weak classifier $\vec{h}_k$

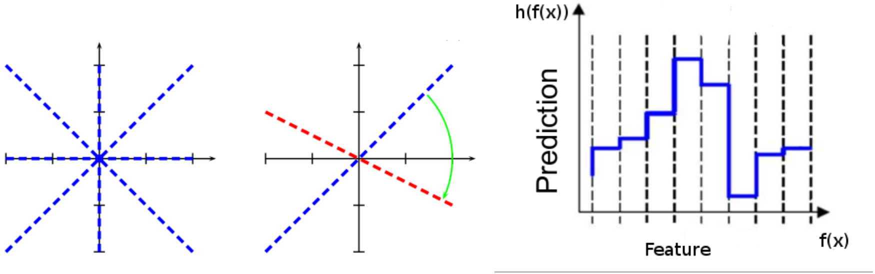


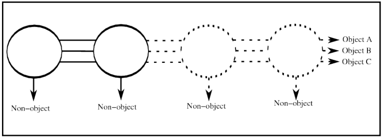

Figure: Left: Joint, Center: Coupled (Proposed), Right: domain partitioning

# Optimization problem

| Method | Problem | Solution | N. of Prob. | N. of Variables | Order $O()$ |
|---|---|---|---|---|---|
| Adaboost | $\min_{c_j} \sum_j w_+^j e^{-c_j} + w_-^j e^{+c_j}$ | Analitic | $J$ | $J$ | $N + J$ |
| Joint | $\min_{c_j, b_m \in \{0,1\}} \sum_{j,m} w_+^{j,m} e^{-\beta_m c_j} + w_-^{j,m} e^{\beta_m c_j}$ | Analytic | $2^M$ | $J(+M)$ | $N + J2^M$ |
| Independent | $\min_{c_{j,m}} \sum_m \sum_j w_+^{j,m} e^{-c_{j,m}} + w_-^{j,m} e^{c_{j,m}}$ | Analytic | $J$ | $JM$ | $N + JM$ |
| Coupled | $\min_{c_j, \gamma_m \in \mathbb{R}} \sum_{j,m} w_+^{j,m} e^{-\gamma_m c_j} + w_-^{j,m} e^{\gamma_m c_j}$ | Newton * | 1 | $J + M$ | $N + (J + M)^2$ |

Weak classifier: domain partitioning ($J$ bins); $M$ classes; $N$ training examples

# Multiclass cascade

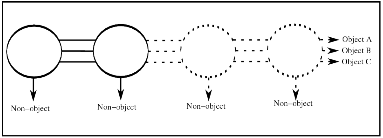



## Multiclass boosted classifier

$$\vec{H}(x) = \sum_{t=0}^{T} \vec{h}_t(f_t(x))$$

Condition to verify at each layer: $\forall \vec{V} \in \mathbf{V}_m, \vec{V} \cdot \vec{H}(x) \geq 0$

# Multiclass cascade




## Multiclass boosted classifier

$$\vec{H}(x) = \sum_{t=0}^{T} \vec{h}_t(f_t(x))$$

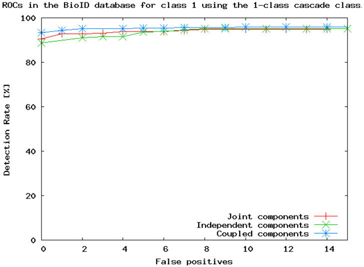
Condition to verify at each layer: $\forall \vec{V} \in \mathbf{V}_m, \vec{V} \cdot \vec{H}(x) \geq 0$

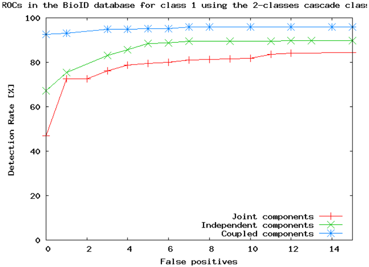
## Multiclass nested cascade

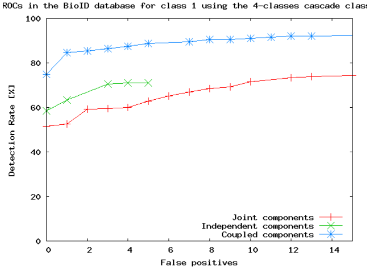
$$\vec{H}_k(x) = \vec{H}_{k-1}(x) + \sum_{t=0}^{T} \vec{h}_{t,k}(f_{t,k}(x)), \text{ with } \vec{H}_0(x) = 0$$

# Results: multiclass weak classifier

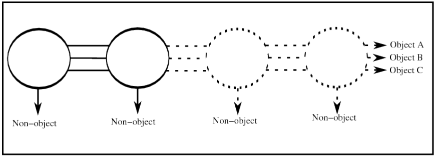
1, 2 and 4 classes when using Independent, Joint and Coupled weak classifiers.



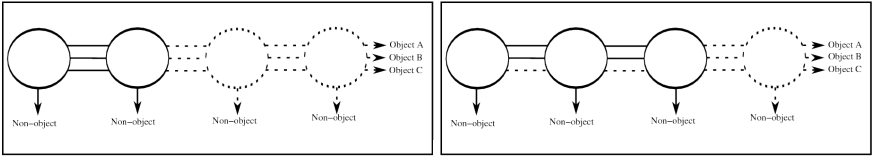



# Coarse-To-fine (CTF) Multiclass Cascade



# Coarse-To-fine (CTF) Multiclass Cascade




the inequalities $\forall \vec{V} \in \mathbf{V}_m, \vec{V} \cdot \vec{H}(x) \geq 0$ are now verified separately through the cascade.

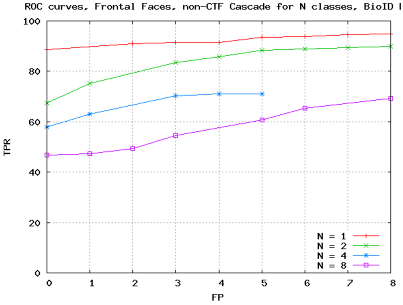
$$\vec{H}_k(x) = \left[ \vec{H}_{k-1}(x) + \sum_{t=1}^{T_k} \vec{h}_{k,t}(f_{k,t}(x)) \right] \odot \vec{\mathbf{A}}_{k-1}(x)$$

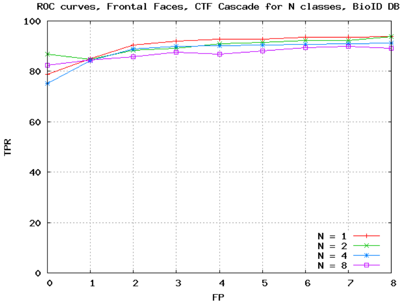
$$\text{with } \vec{H}_0(x) = \vec{0}, \text{ and } \vec{\mathbf{A}}_0(x) = \vec{1} \,,$$

$$\mathbf{A}_k(x,m) = \prod_{i=0}^{k} u\left(H_i(x,m)\right) = u\left(H_k(x,m)\right) \prod_{i=0}^{k-1} \mathbf{A}_i(x,m)$$

$\odot$: point-wise product

# NonCFT (left) vs CTF (right) multiclass cascade

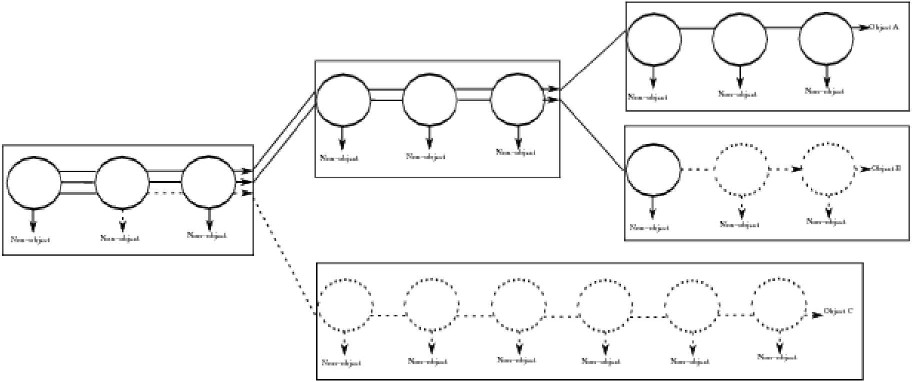


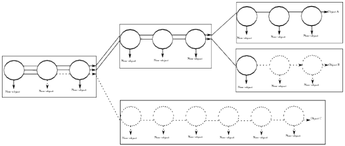

## Average processing time [sec] for a 384x286 pixels image

| Method | Number of classes | | | |
|---|---|---|---|---|
| | 1 | 2 | 4 | 8 |
| NonCTF | 0.22 | 0.4 | 0.94 | 4.46 |
| CTF (log(n)) | 0.22 | 0.33 | 0.60 | 0.67 |

# TCAS: Tree of nested CAScades



# TCAS: Tree of nested CAScades



A node $\mathbb{N}$ of a TCAS $\mathbb{T}$ consist of:

- $n_{\mathbb{N}}$ siblings, $\{\mathbb{N}_s\}_{s=\{1,\ldots n_{\mathbb{N}}\}}$, and
- a multiclass nested cascade classifier $\vec{H}_{\mathbb{C}}$,
- a mask $\vec{\mathbf{A}}_{\mathbb{N}} \in \{0,1\}^M$,
- and a "pointer", $p_{\mathbb{N}}$, to its direct ancestor.

The output of a node $\mathbb{N}$ is defined as:

$$\vec{H}_{\mathbb{N}}(x) = \vec{H}_{\mathbb{C}}(x) + \vec{H}_{p_{\mathbb{N}}}(x) \odot \vec{\mathbf{A}}_{\mathbb{N}},$$

with $\vec{H}_{p_{\mathbb{N}}}(x)$ the output of the ancestor of $\mathbb{N}$, $p_{\mathbb{N}}$.

1. Introduction
   - Cascade classifiers
   - Adaboost
2. Multiclass Object Detection
   - Multiclass boosting and weak classifiers
   - Multiclass cascade
   - TCAS: Tree of nested CAScades
3. Results, conclusions and future work

# Results

## Three Problems

- Multiview face detection (in-plane-rotation)
- Frontal Face, left fist (hand) and right fist (hand) detection
- Humanoid and multiview Aibo robot detection

# Results

## Three Problems

- Multiview face detection (in-plane-rotation)
- Frontal Face, left fist (hand) and right fist (hand) detection
- Humanoid and multiview Aibo robot detection

## Evaluation

Compare the use of a TCAS with the use of parallel cascade classifiers

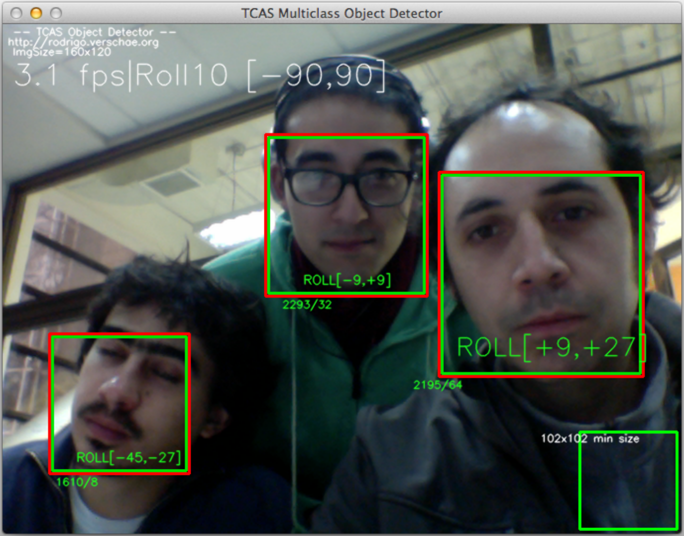
- Detection rate vs FP (ROC curves)
- Processing time

# Results

# Results: Face Detection (in-plane rotations)

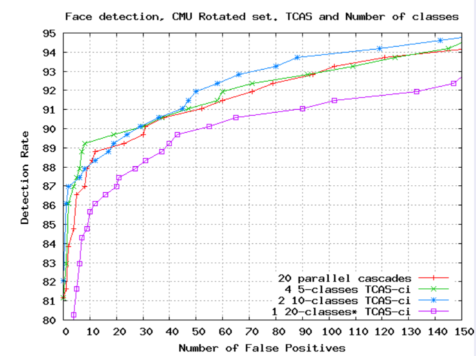
# Results: Face Detection (in-plane rotations)

# Results: Face Detection (in-plane rotations)



# Results Face Detection (in-plane rotations)

## Accuracy (in the CMU Rotated Set): increasing the number of classes




Processing time compared to 20 parallel cascades:

- 4 TCAS (5-classes) is ∼2.3 times faster
- 2 TCAS (10-classes) is ∼2.4 times faster
- 1 TCAS (20-classes) is ∼1.6 times faster

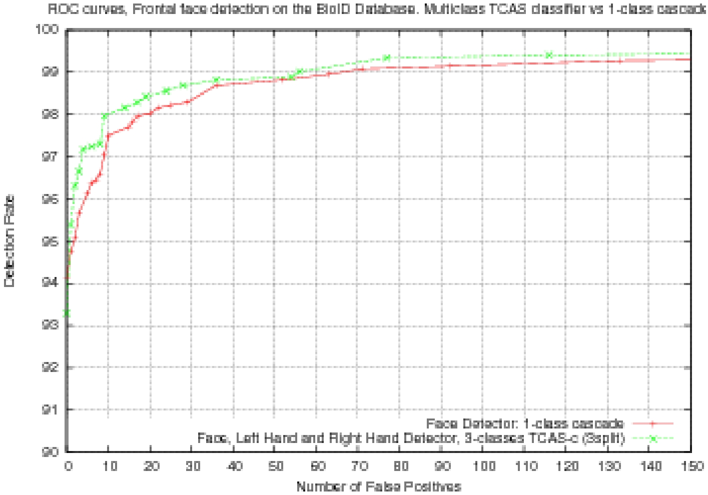
# Results: Hand and Face detection

# Results: Hand and Face detection

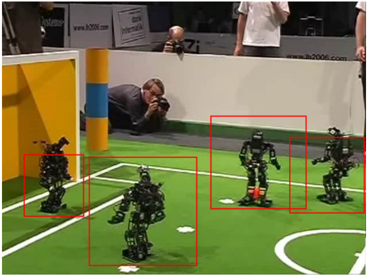
Table: Average Processing times [sec] of frontal hand (Right and Left fist) and frontal face detector(s) on the CMU database.

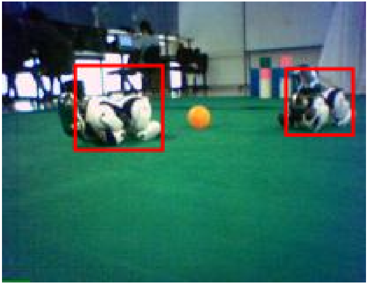
| Classifier | Classes | DB: CMU |
|---|---|---|
| Three 1-class cascades | Face & Hands | 1.27 |
| 1-class + 2-classes tcas | Face & (Left & Right Hand) | 0.96 |
| 1-class + 2-classes tcas | Left & (Right Hand & Face) | 1.05 |
| 3-classes cascade | Face & Hands | – |
| 3-classes tcas (split3) | Face & Hands | **0.78** |

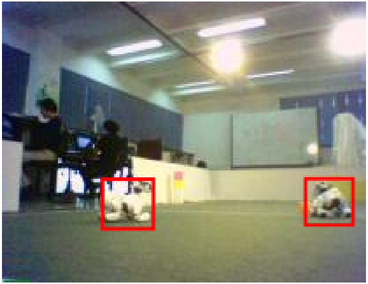
# Results: Hand and Face detection



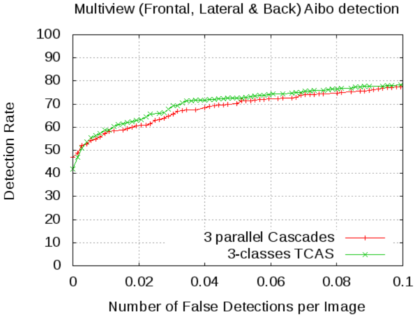
# Results: Robot detection

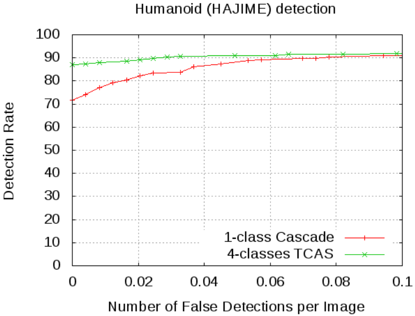






# Results: Robot detection



# Conclusions

## TCAS: a CTF nested tree of Multiclass Cascades

- CTF search in the negative target space (cascade-like)
- CTF search in the positive target space (CTF tree-like)
- Object Detection for multiview and multiclass problems

# Conclusions

## TCAS: a CTF nested tree of Multiclass Cascades

- CTF search in the negative target space (cascade-like)
- CTF search in the positive target space (CTF tree-like)
- Object Detection for multiview and multiclass problems

## Results

- Applied to 3 problems:
  - Multiview face detection
  - Multiclass and Multiview Robot detection, and
  - Hand and Face detection.
- TCASs are faster and more accurate than using parallel cascades.
- The processing time is $\log(n)$ in the number of classes $n$.
- The training time grows linearly with the number of training examples
- Training takes less than a day on a modern computer.

# Current and Future work

## Classifier

- class grouping (based on feature location)
- fine view estimation ("regression")

## Extension

- feature selection per layer/node of the structure
- scale up in the number of classes/views
- more object classes and DBs
- other sensors (RGB-D, Thermal, etc.)

# Thank you for your attention

## Any questions?

The presentation will be available on my webpage soon

## Contact

- `rodrigo@verschae.org`
- `http://rodrigo.verschae.org`