Target Detection in Images

Michael Elad*

Scientific Computing and Computational Mathematics

Stanford University

High Dimensional Data Day

February 21th, 2003

* Joint work with **Yacov Hel-Or** (IDC, Israel),

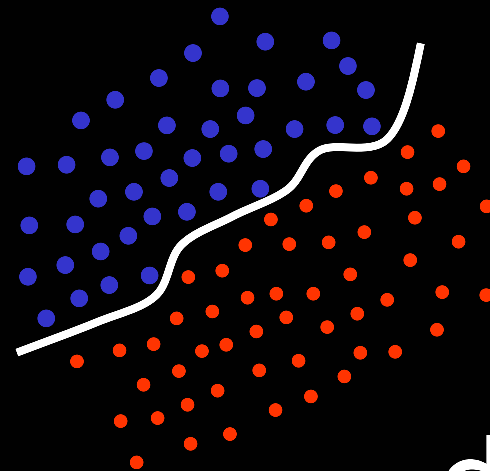
and **Renato Keshet** (HPL-Israel).

Part 1

Why Target Detection is Different ?

1. High Dimensional Data

- Consider a cloud of d-dimensional data points.
- Classic objective – Classification: separate the cloud of points into several sub-groups, based on labeled examples.
- Vast amount of literature about how to classify – Neural-Nets, SVM, Boosting, …
  - These methods are 'too' general,
  - These methods are 'blind' to the clouds structure,
  - What if we have more information?



$\Re^d$

2. Target Detection

Input image

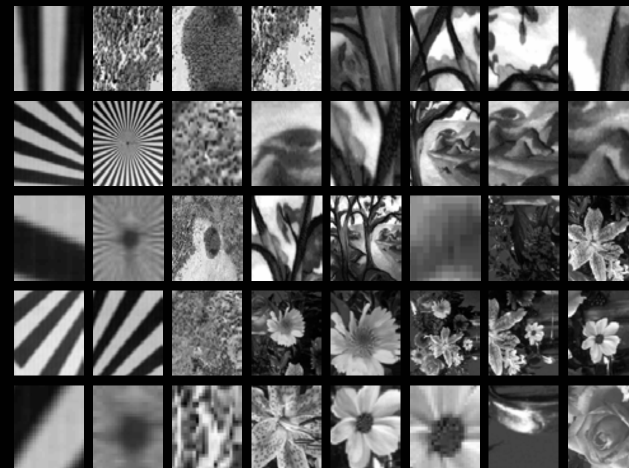
Target (Face) Detector

Output image

Claim: Target detection in images is a classification problem for which we have more information:

- The d-dimensional points are blocks of $\sqrt{d} \times \sqrt{d}$ pixels from the image in **EACH** location and scale (e.g. d≈400).
- Every such block is either *Target* (face) or *Clutter*. The classifier needs to decide which is it.

3. Our Knowledge



Property 1: Volume{ *Target* } << Volume{ *Clutter* }.

Property 2: Prob{ *Target* } << Prob{ *Clutter* }.

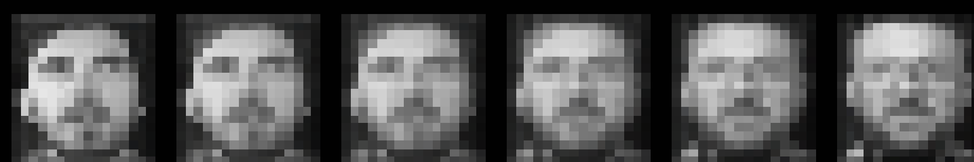
Property 3: *Target* = sum of **few** convex sub-groups.

4. Convexity - Example

Is the *Faces* set is convex?

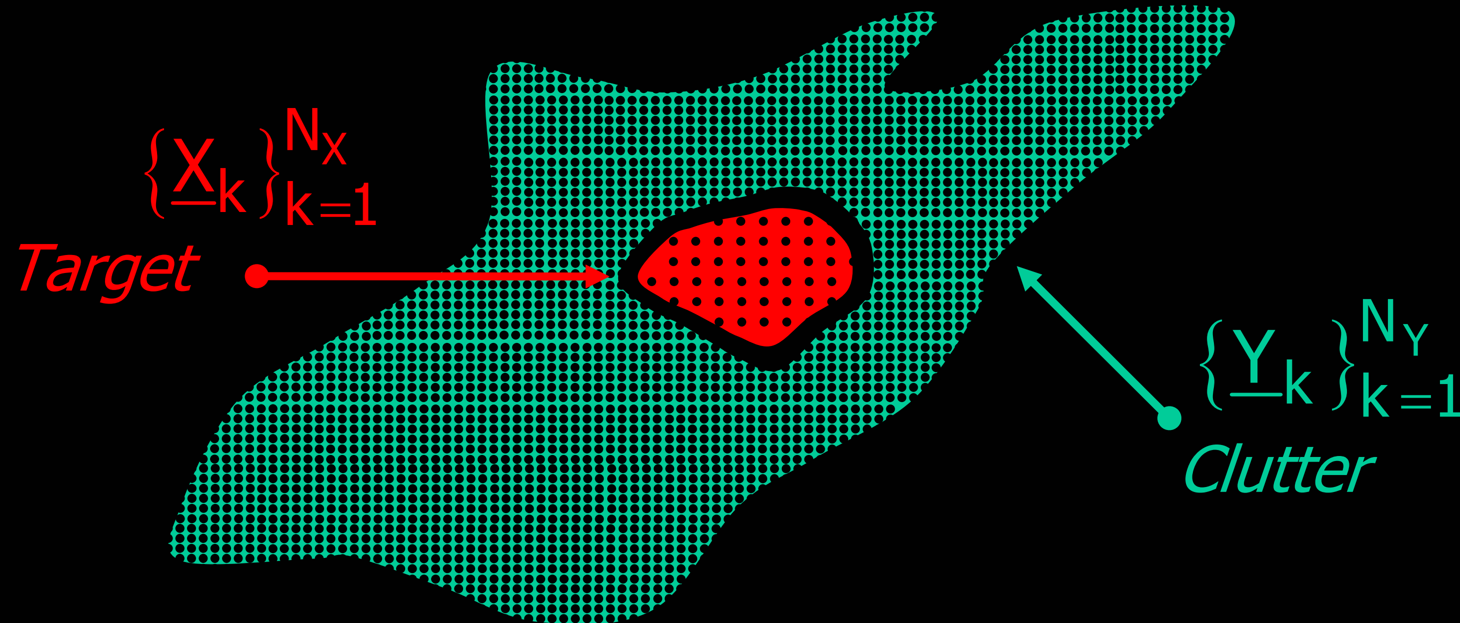
- Frontal and vertical faces

- A low-resolution representation of the faces



- For rotated faces, slice the class into few convex sub-groups.

5. Our assumptions



- Volume{*Target*} << Volume{*Clutter*}
- Prob{*Target*} << Prob{*Clutter*}.
- Simplified: The *Target* class is (nearly) convex.

6. The Objective

Design of a classifier of the form

$$C\{\underline{Z},\underline{\theta}\}:\ \Re^{d}\times\Re^{J}\rightarrow\{-1,+1\}$$

Need to answer three questions:

Q1: What parametric form to use? Linear or non-linear? What kind of non-linear?

Q2: Having chosen the parametric form, how do we find appropriate set of parameters $\underline{\theta}$?

Q3: How can we exploit the properties we have mentioned before in answering Q1 and Q2 smartly?

Part 2

SOME

Previous Work on Face Detection

1. Neural Networks

- Choose C(Z,θ) to be a Neural Network (NN).
- Add prior knowledge in order to:
  - Control the structure of the net,
  - Choose the proper kind (RBF ?),
  - Pre-condition the data (clustering)
- Representative Previous Work:
  - Juel & March (1996), and
  - Rowley & Kanade (1998), and
  - Sung & Poggio (1998).

NN leads to a
Complex Classifier

2. Support Vector Machine

- Choose C(Z,θ) to be a based on SVM.
- Add prior knowledge in order to:
  - Prune the support vectors,
  - Choose the proper kind (RBF, Polynomial ?),
  - Pre-condition the data (clustering)
- Similar story applies to Boosting methods.
- Representative Previous Work:
  - Osuna, Freund, & Girosi (1997),
  - Bassiou et.al.(1998),
  - Terrillon et. al. (2000).

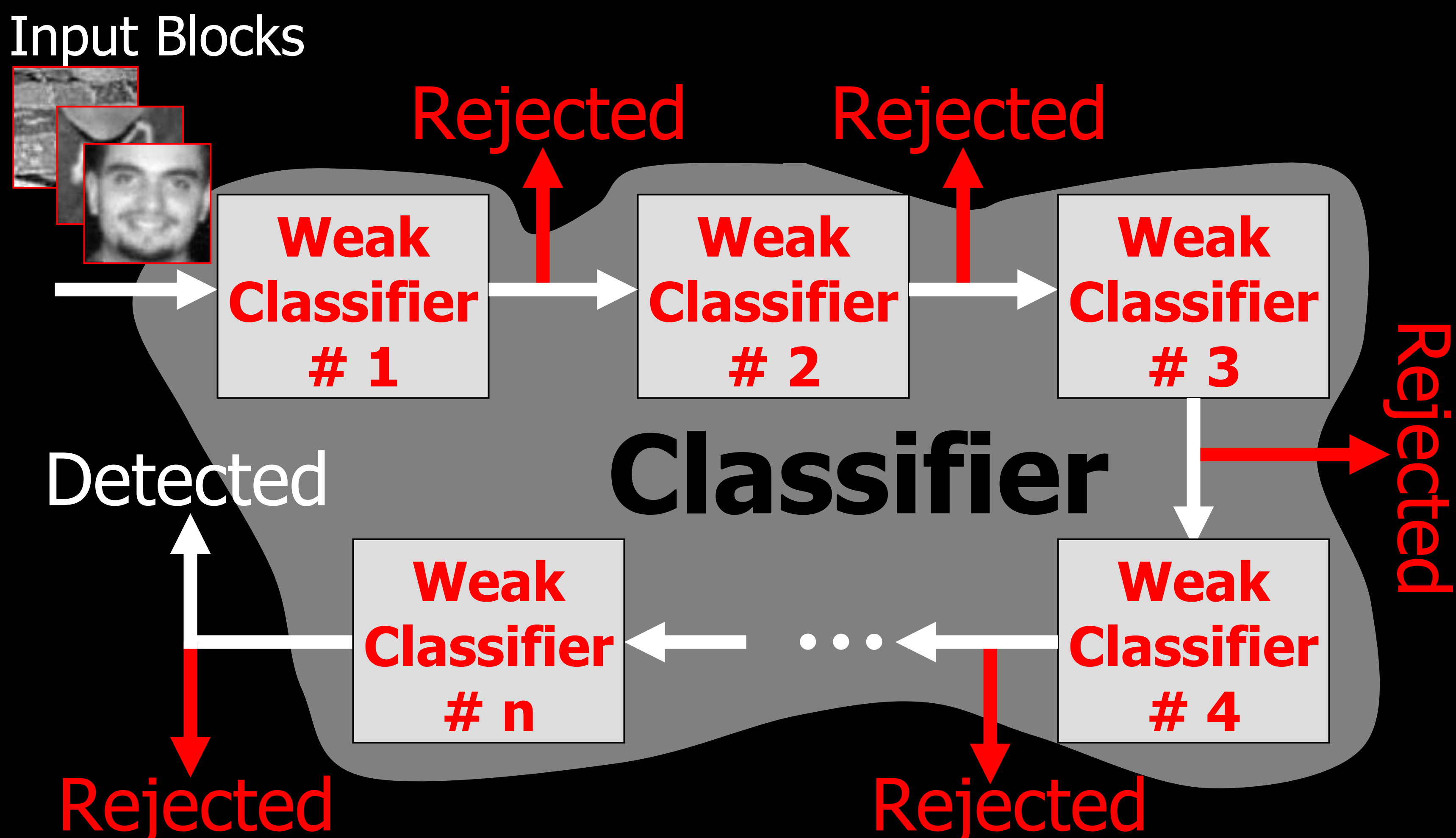
SVM leads to a Complex Classifier

3. Rejection Based

- Build C($\underline{Z}$,$\underline{\theta}$) as a combination of weak (simple to design and activate) classifiers.
- Apply the weak classifiers sequentially while rejecting non-faces.
- Representative Previous Work:
  - Rowley & Kanade (1998)
  - Elad, Hel-Or, & Keshet (1998),
  - Amit, Geman & Jedyank (1998),
  - Osdachi, Gotsman & Keren (2001), and
  - Viola & Jones (2001).

Fast
(and accurate) classifier

4. The Rejection Idea



5. Supporting Theory

- (Ada) Boosting – Freund & Schapire (1990-2000) – Using a group of weak classifiers in order to design a successful complex classifier.
- Decision-Tree – Tree structured classification (the rejection approach here is a simple dyadic tree).
- Rejection – Nayar & Baker (1995) - Application of rejection while applying the sequence of weak classifiers.
- Maximal Rejection – Elad, Hel-Or & Keshet (1998) – Greedy approach towards rejection.

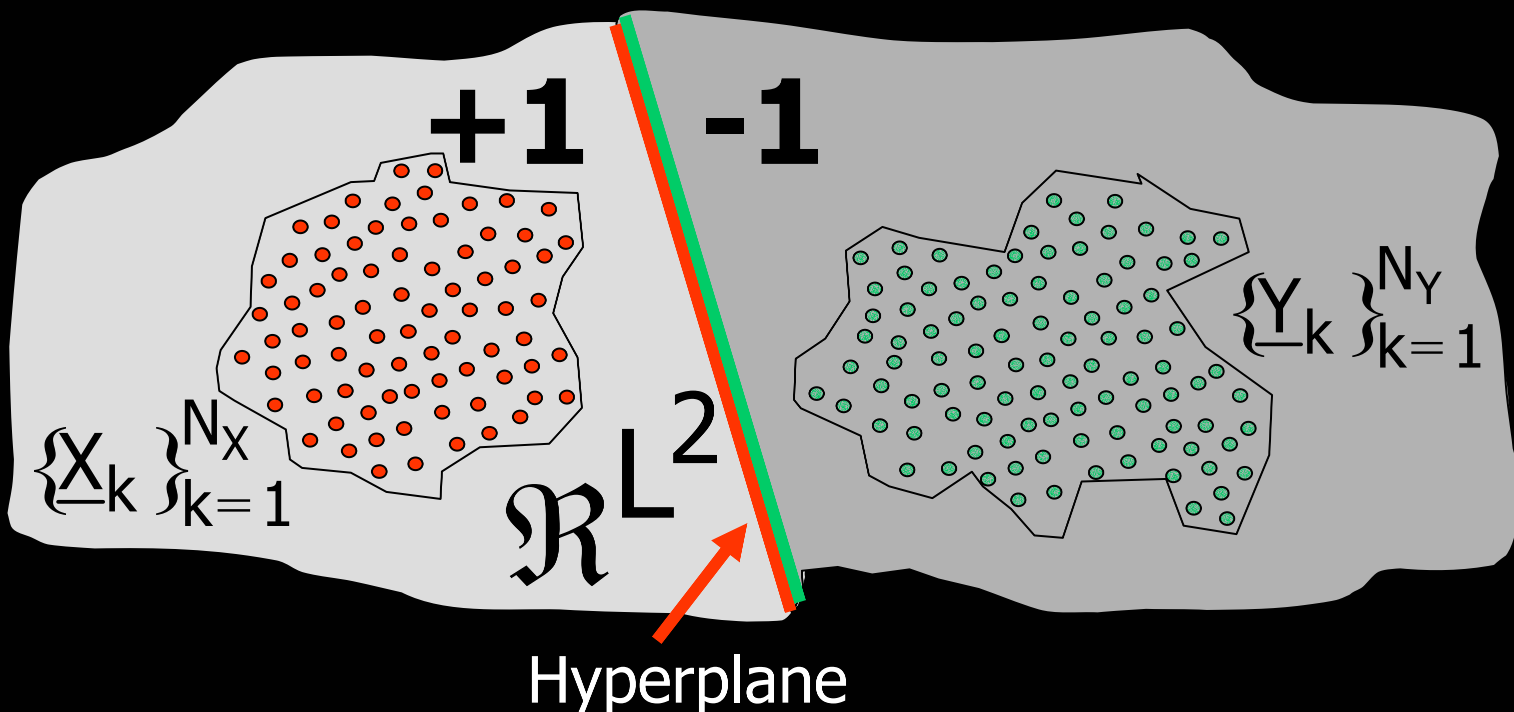
Part 3

Maximal Rejection Classification

1. Linear Classification (LC)

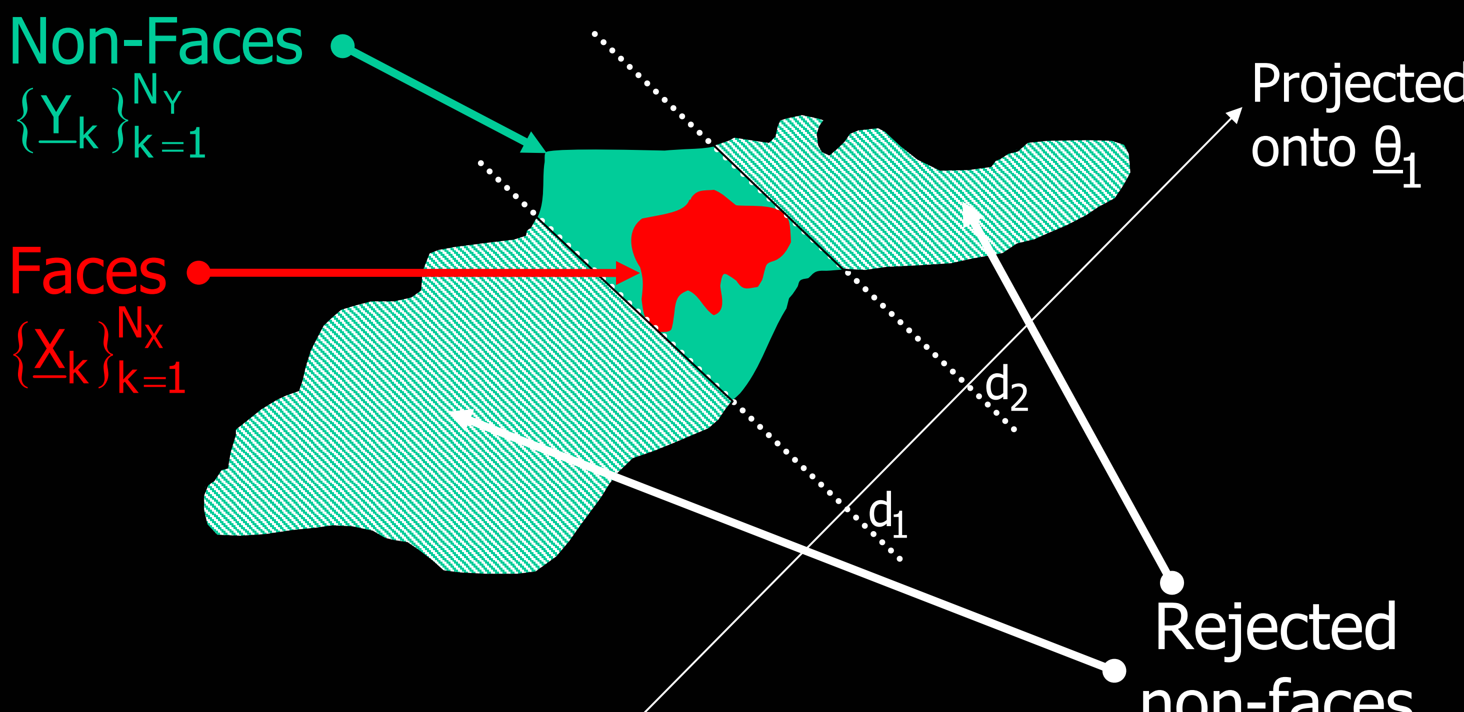
We propose LC as our weak classifier:

$$C\{\underline{Z},\underline{\theta}\} = \text{sign}\left\{\underline{Z}^T\underline{\theta} - \theta_0\right\}$$



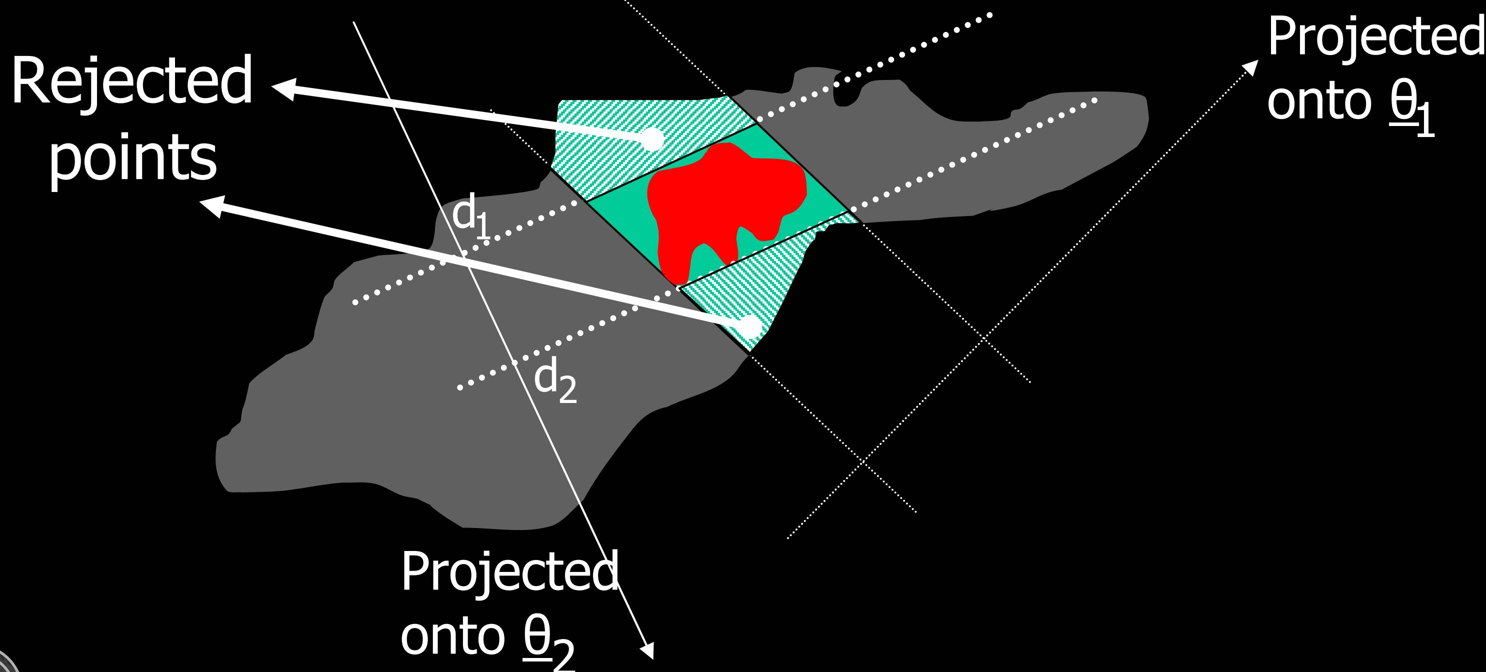
2. Maximal Rejection

Find $\underline{\theta}_1$ and two decision levels $\left[d_1, d_2\right]_1$ such that the number of rejected non-faces is maximized while finding all faces

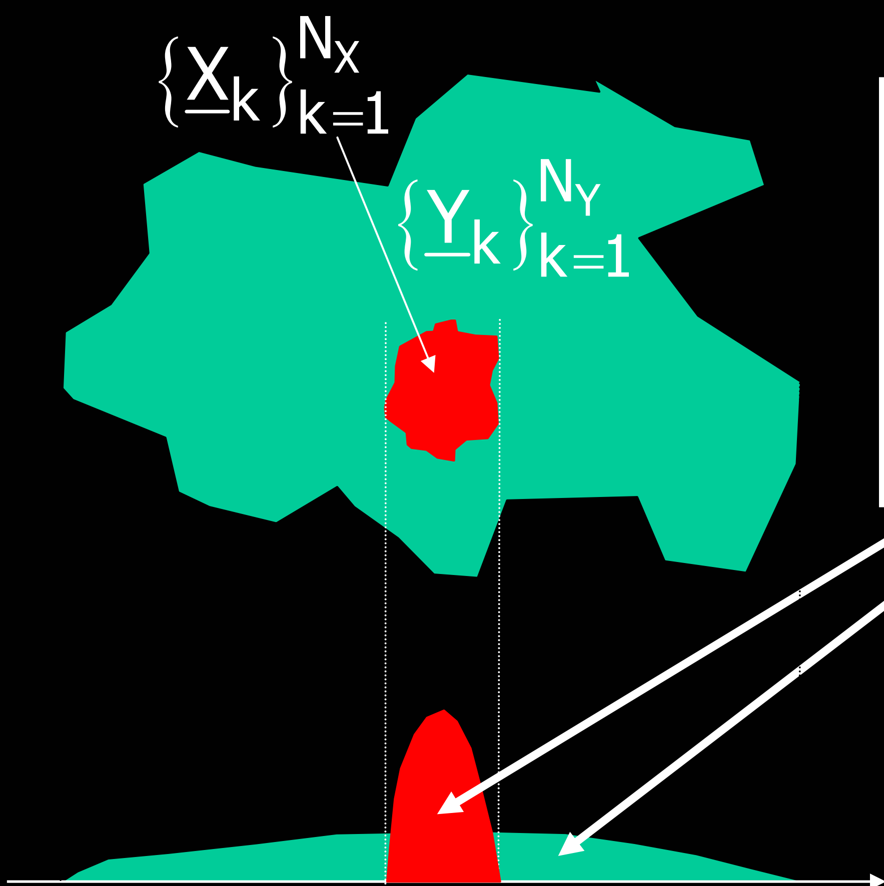


3. Iterations

Taking ONLY the remaining non-faces:
Find $\underline{\theta}_2$ and two decision levels $\left[d_1, d_2\right]_2$ such that the number of rejected non-faces is maximized while finding all faces



4. Maximizing Rejection



Maximal Rejection

⇕

Maximal distance between these two PDF's

⇓

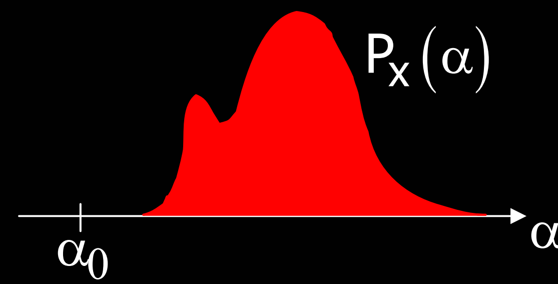
We need a measure for this distance which will be appropriate and easy to use

5. One Sided Distance

Define a distance between a point and a PDF by

$$D_1\left\{\alpha_0, P_x(\alpha)\right\} = \int_{\alpha} \frac{(\alpha_0-\alpha)^2}{r_x^2} P_x(\alpha)\, d\alpha$$

$$= \frac{(\alpha_0 - m_x)^2 + r_x^2}{r_x^2}$$



$$D_2\left\{ P_x(\alpha), P_y(\alpha)\right\} = \int_{\alpha} D_1\left\{\alpha, Px(\alpha)\right\} P_y(\alpha) d\alpha = \frac{(m_x - m_y)^2 + r_x^2 + r_y^2}{r_x^2}$$

This distance is asymmetric !! It describes the average distance between points of Y to the X-PDF, $P_X(\alpha)$.

6. Final Measure

$$D_3\{P_x(\alpha),P_y(\alpha)\}=P(Y)\frac{(m_x-m_y)^2+r_x^2+r_y^2}{r_x^2}+P(X)\frac{(m_x-m_y)^2+r_x^2+r_y^2}{r_y^2}$$

In the case of face detection in images we have

P(X)<<P(Y)

We Should Maximize

(GEP)

$$f\{\underline{\theta}\}=\frac{\underline{\theta}^T\left\{\left[\underline{M}_X-\underline{M}_Y\right]\left[\underline{M}_X-\underline{M}_Y\right]^T+R_X+R_Y\right\}\underline{\theta}}{\underline{\theta}^T R_X\underline{\theta}}$$

7. Different Method 2

Maximize the following function:

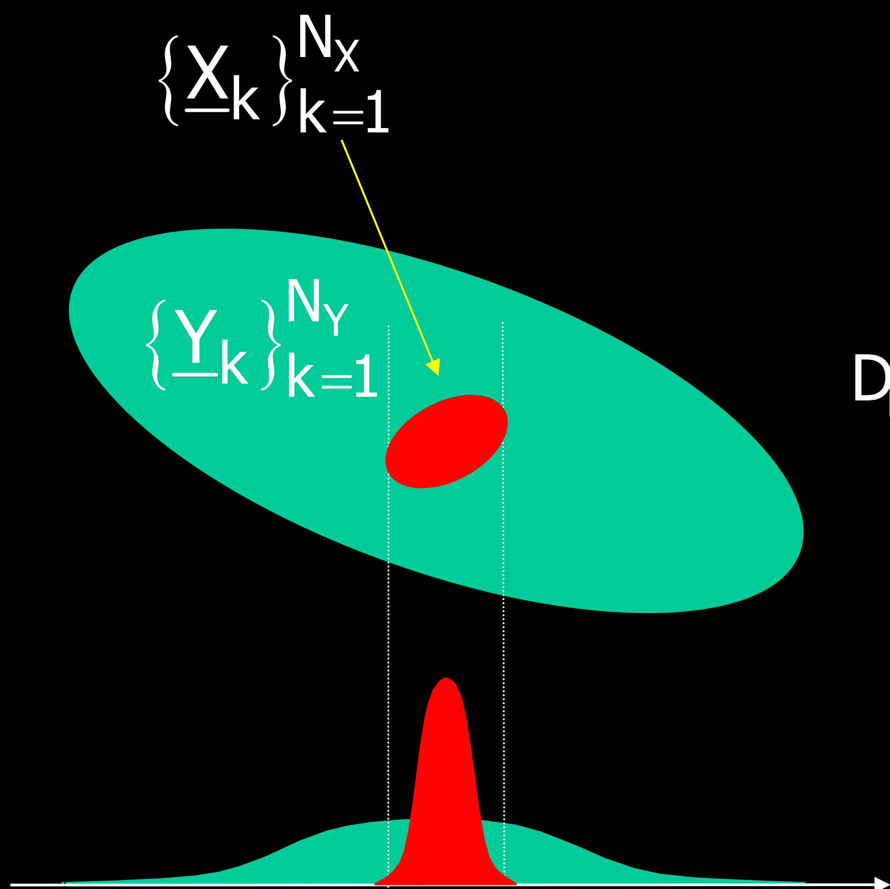
Maximize the distance between all the pairs of [face, non-face]

$$f\{\underline{\theta}\} = \frac{\sum_{j=1}^{N_Y}\sum_{k=1}^{N_X}\left[\underline{\theta}^T\underline{X}_k - \underline{\theta}^T\underline{Y}_j\right]^2}{\sum_{j=1}^{N_X}\sum_{k=1}^{N_X}\left[\underline{\theta}^T\underline{X}_k - \underline{\theta}^T\underline{X}_j\right]^2} = C\cdot\frac{\underline{\theta}^T\mathbf{R}\underline{\theta}}{\underline{\theta}^T\mathbf{Q}\underline{\theta}} =$$

The same Expression

Minimize the distance between all the pairs of [face, face]

8. Different Method 3



If the two PDF's are assumed Gaussians, their KL distance is given by

$$D_{KL}\{P_x,P_y\}=\frac{(m_x-m_y)^2+r_x^2+r_y^2}{2r_x^2}+\ln\left\{\frac{r_x}{r_y}\right\}-1$$

And we get a similar expression

9. Back to Our Assumptions

- Volume{*Target*} << Volume{*Clutter*}:
  Sequential rejections succeed because of this property.
- Prob{*Target*} << Prob{*Clutter*}:
  Speed of classification is guaranteed because of this property.
- The *Target* class is nearly convex:
  Accuracy (low P_F and high P_D) is emerging from this property

The MRC algorithm idea is strongly dependent on these assumptions, and it leads to

Fast & Accurate Classifier.

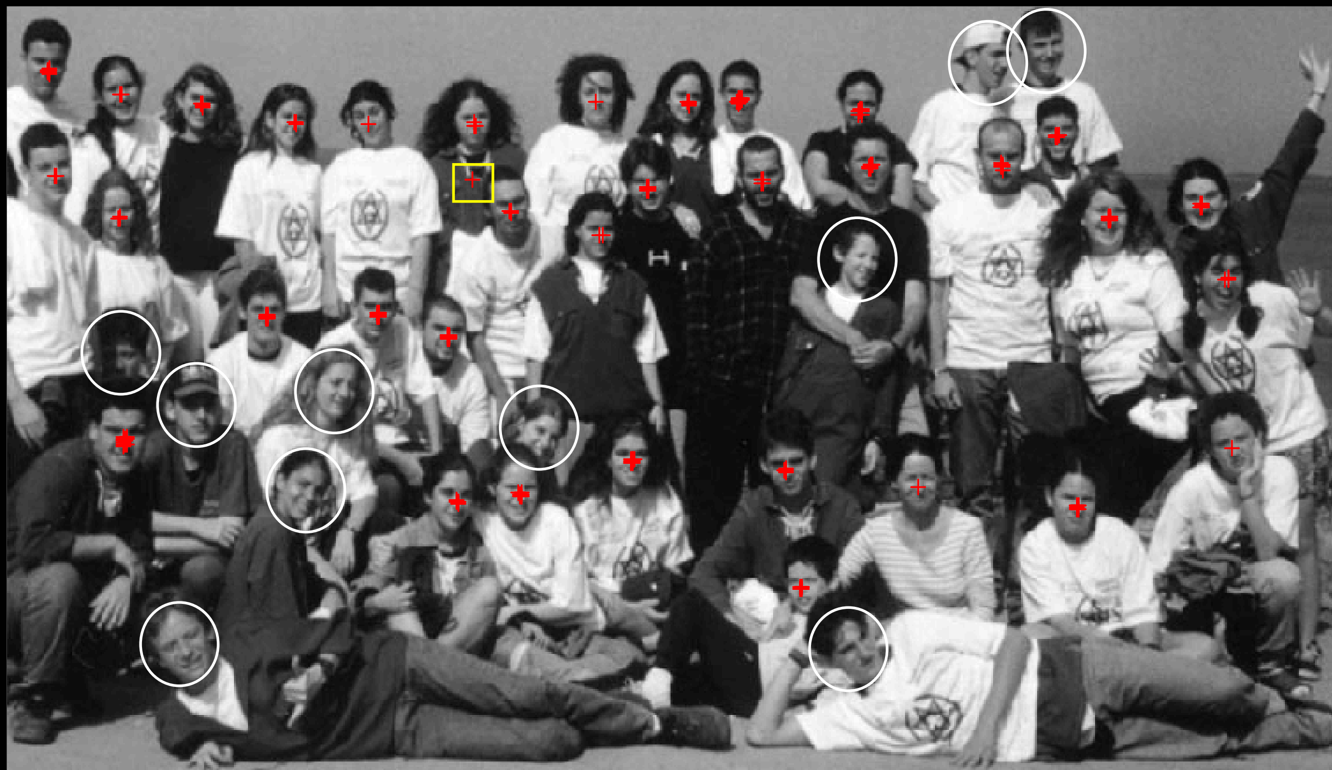
Chapter 4

Results & Conclusions

1. Experiment Details

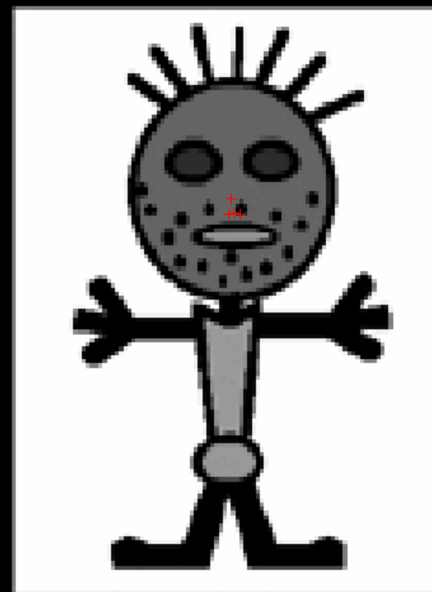
- Kernels for finding *faces* (15·15) and *eyes* (7·15).
- Searching for eyes and faces sequentially - very efficient!
- Face DB: 204 images of 40 people (ORL-DB after some screening). Each image is also rotated ±5° and vertically flipped - to produce 1224 Face images.
- Non-Face DB: 54 images - All the possible positions in all resolution layers and vertically flipped - about $40 \cdot 10^6$ non-face images.
- Core MRC applied (no second layer, no clustering).

2. Results - 1



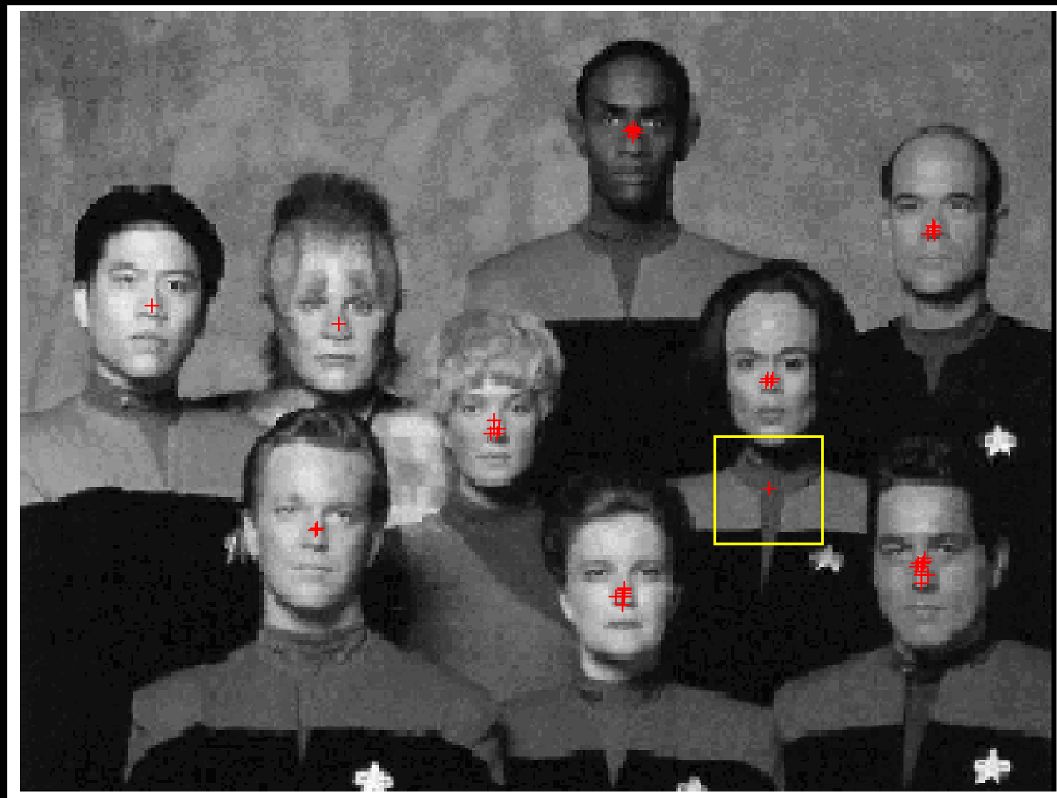
Out of 44 faces, 10 faces are undetected, and 1 false alarm
(the undetected faces are circled - they are either rotated or strongly shadowed)

3. Results - 2



All faces detected with no false alarms

4. Results - 3



All faces detected with 1 false alarm

(looking closer, this false alarm can be considered as face)

5. More Details

- A set of 15 kernels - the first typically removes about 90% of the pixels from further consideration. Other kernels give an average rejection of 50%.
- The algorithm requires slightly more that one **convolution** of the image (per each resolution layer).
- Compared to state-of-the-art results:
  - Accuracy – Similar to Rowley and Viola.
  - Speed – Similar to Viola – much faster (factor of ~10) compared to Rowley.

6 .Conclusions

- Rejection-based classification - effective and accurate.
- Basic idea – group of weak classifiers applied sequentially followed each by rejection decision.
- Theory – Boosting, Decision tree, Rejection based classification, and MRC.
- The Maximal-Rejection Classification (MRC):
  - Fast – in close to one convolution we get detection,
  - Simple – easy to train, apply, debug, maintain, and extend.
  - Modular – to match hardware/time constraints.
  - Limitations – can be overcome.
- More details – http://www-sccm.stanford.edu/~elad

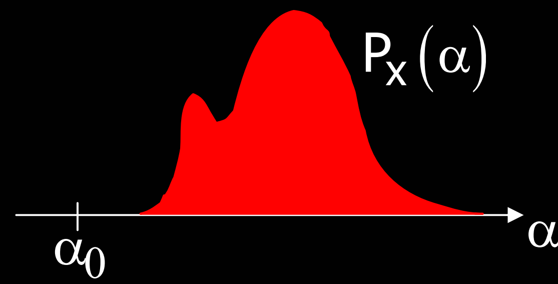
7 . More Topics

1. Why scale-invariant measure?
2. How we got the final distance expression?
3. Relation of the MRC to Fisher Linear Discriminant
4. Structure of the algorithm
5. Number of convolutions per pixel
6. Using color
7. Extending to 2D rotated faces
8. Extension to 3D rotated faces
9. Relevancy to target detection
10. Additional ingredients for better performance
11. Design considerations

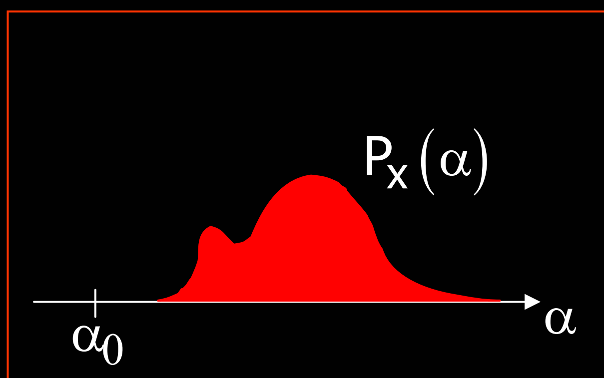
1. Scale-Invariant

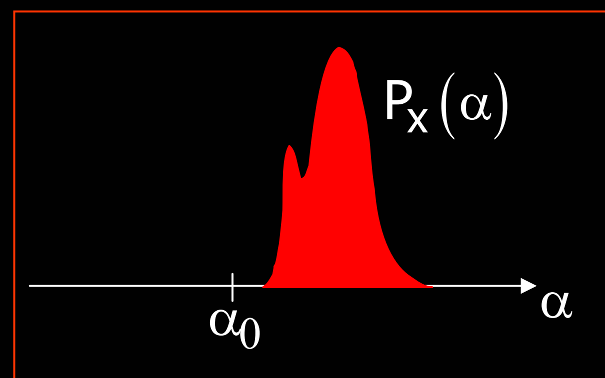
$$D_1\left\{\alpha_0, P_x(\alpha)\right\} = \int_\alpha \frac{(\alpha_0-\alpha)^2}{r_x^2} P_x(\alpha)\,d\alpha$$

$$= \frac{(\alpha_0 - m_x)^2 + r_x^2}{r_x^2}$$



Same distance for





$$f\{\underline{\theta}\} = \frac{\underline{\theta}^T \left\{ [\underline{M}_X - \underline{M}_Y][\underline{M}_X - \underline{M}_Y]^T + R_X + R_Y \right\} \underline{\theta}}{\underline{\theta}^T R_X \underline{\theta}}$$

In this expression:

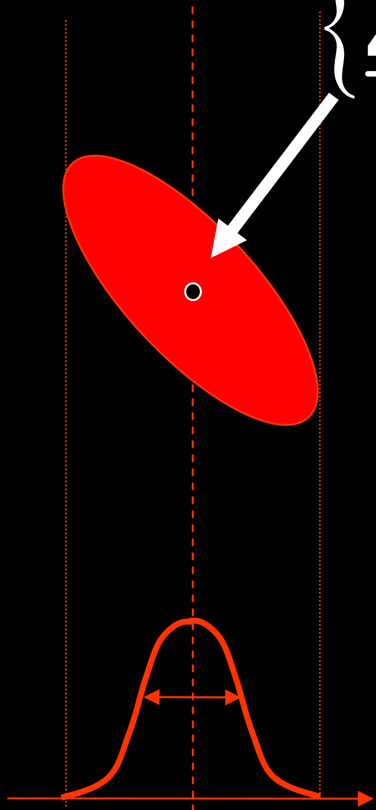
1. The two classes means are encouraged to get far from each other
2. The Y-class is encouraged to spread as much as possible, and
3. The X-class is encouraged to condense to a near-constant value

Thus, getting good rejection performance.

back

2. The Distance Expression

$$\left\{\underline{Z}_k\right\}_{k=1}^{N}$$



$$\underline{M} = \frac{1}{N}\sum_{k=1}^{N} \underline{Z}_k$$

$$\mathbf{R} = \frac{1}{N}\sum_{k=1}^{N} \left[\underline{Z}_k - \underline{M}\right]\left[\underline{Z}_k - \underline{M}\right]^T$$

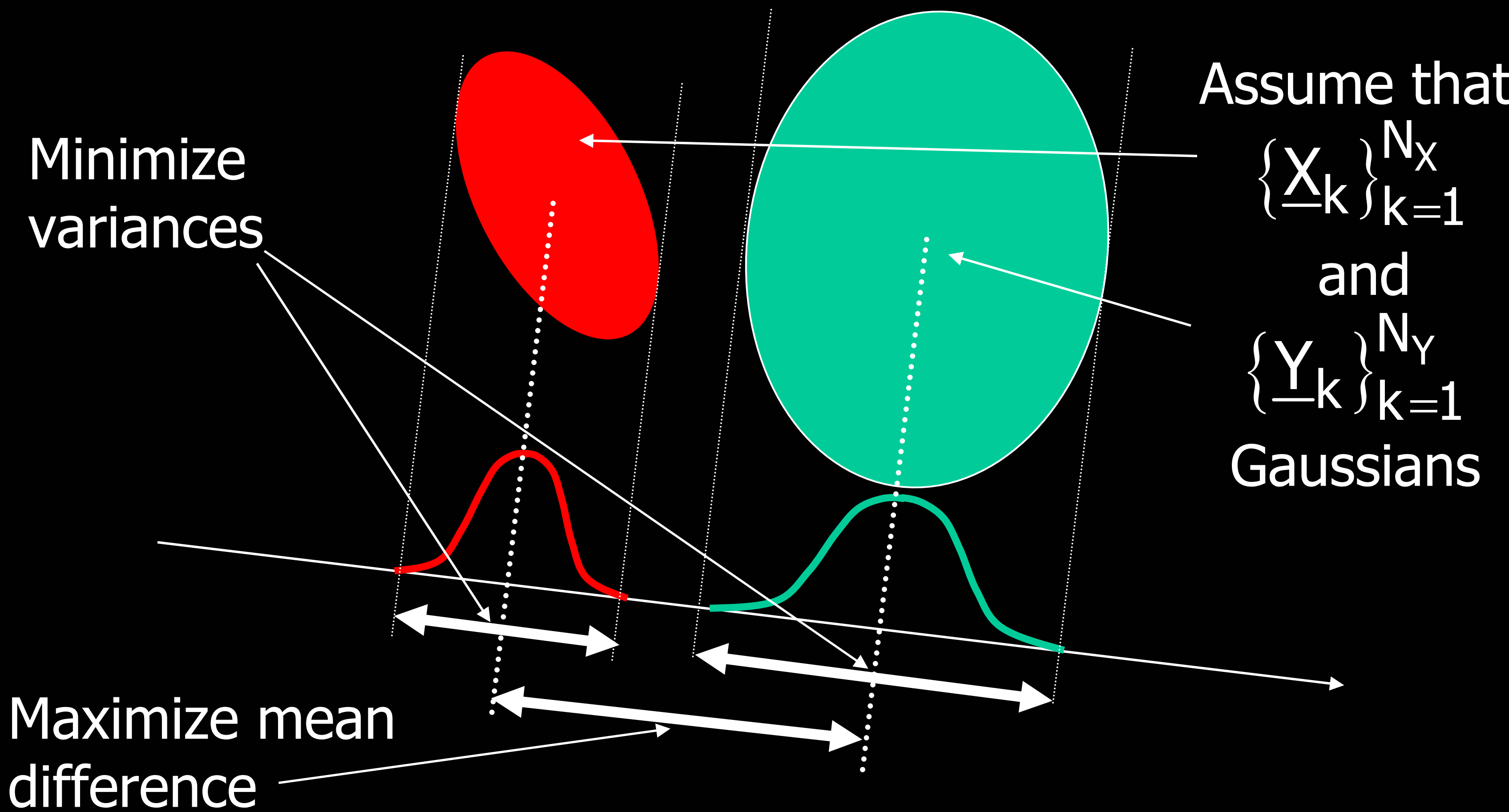
$$z_k = \underline{\theta}^T \underline{Z}_k \Rightarrow m = \underline{\theta}^T \underline{M}, \;\; r = \underline{\theta}^T \mathbf{R}\underline{\theta}$$

$$\frac{(m_y - m_x)(m_y - m_x)^T + r_y^2}{r_x^2} =$$

$$= \frac{\underline{\theta}^T \left[(\underline{M}_y - \underline{M}_x)(\underline{M}_y - \underline{M}_x)^T + \mathbf{R}_y \right] \underline{\theta}}{\underline{\theta}^T \mathbf{R}_x \underline{\theta}}$$

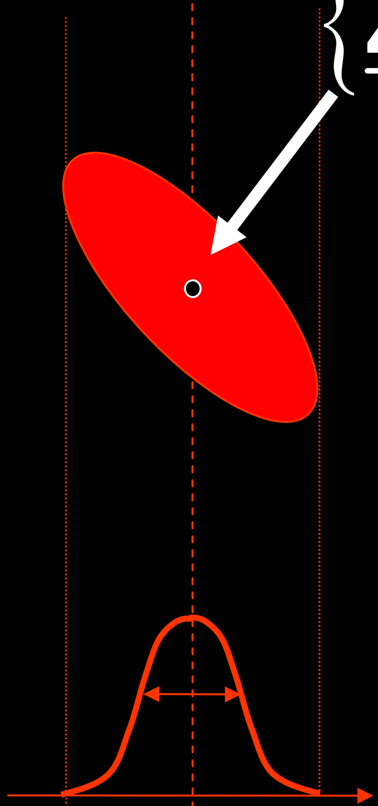
back

3. Relation to FLD*



*FLD - Fisher Linear Discriminant

$$\{\underline{Z}_k\}_{k=1}^{N}$$



$$\underline{M} = \frac{1}{N}\sum_{k=1}^{N} \underline{Z}_k$$

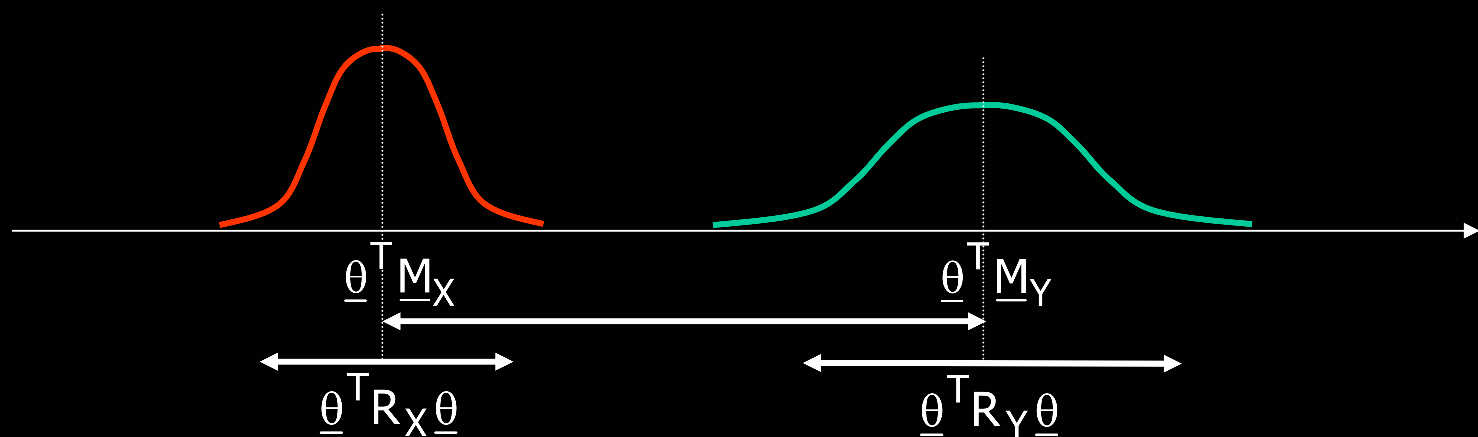
$$\mathbf{R} = \frac{1}{N}\sum_{k=1}^{N} \left[\underline{Z}_k - \underline{M}\right]\left[\underline{Z}_k - \underline{M}\right]^T$$

$$z_k = \underline{\theta}^T \underline{Z}_k \Rightarrow m = \underline{\theta}^T \underline{M}, \quad r = \underline{\theta}^T \mathbf{R} \underline{\theta}$$

Maximize

$$f\{\underline{\theta}\} = \frac{\left[\underline{\theta}^T \underline{M}_X - \underline{\theta}^T \underline{M}_Y\right]^2}{\underline{\theta}^T R_X \underline{\theta} + \underline{\theta}^T R_Y \underline{\theta}} = \frac{\underline{\theta}^T \left[\underline{M}_X - \underline{M}_Y\right]\left[\underline{M}_X - \underline{M}_Y\right]^T \underline{\theta}}{\underline{\theta}^T \left[R_X + R_Y\right]\underline{\theta}}$$

Minimize



In the MRC we got the expression for the distance

$$P(Y)\frac{(m_x - m_y)^2 + r_x^2 + r_y^2}{r_x^2} + P(X)\frac{(m_x - m_y)^2 + r_x^2 + r_y^2}{r_y^2}$$

If P(X)=P(Y)=0.5 we maximize

$$\boxed{\frac{(m_x - m_y)^2 + r_x^2 + r_y^2}{r_x^2}} + \boxed{\frac{(m_x - m_y)^2 + r_x^2 + r_y^2}{r_y^2}}$$

The distance of the Y points to the X-distribution

The distance of the X points to the Y-distribution

Instead of maximizing the sum

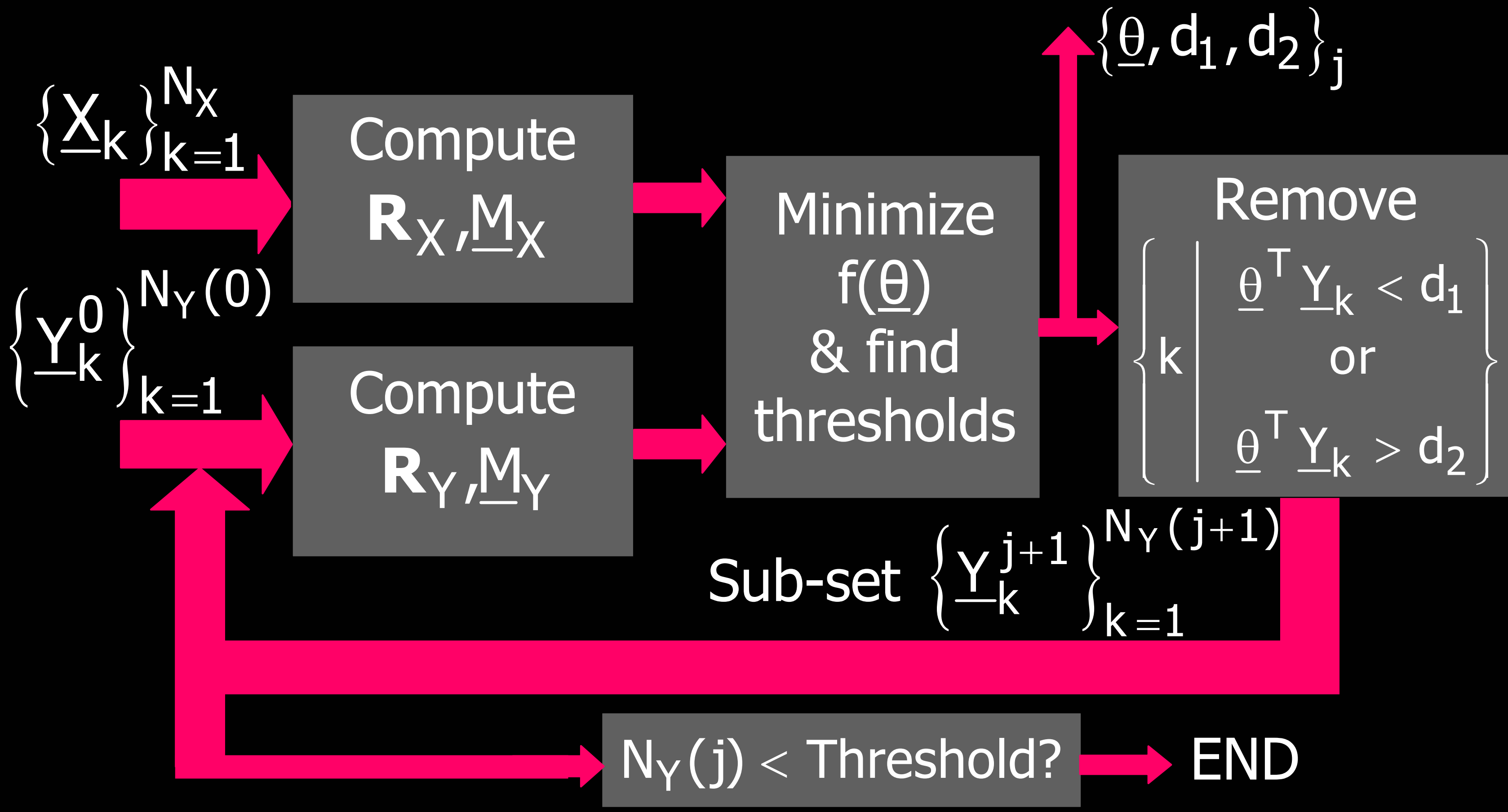
$$\frac{(m_x - m_y)^2 + r_x^2 + r_y^2}{r_x^2} + \frac{(m_x - m_y)^2 + r_x^2 + r_y^2}{r_y^2}$$

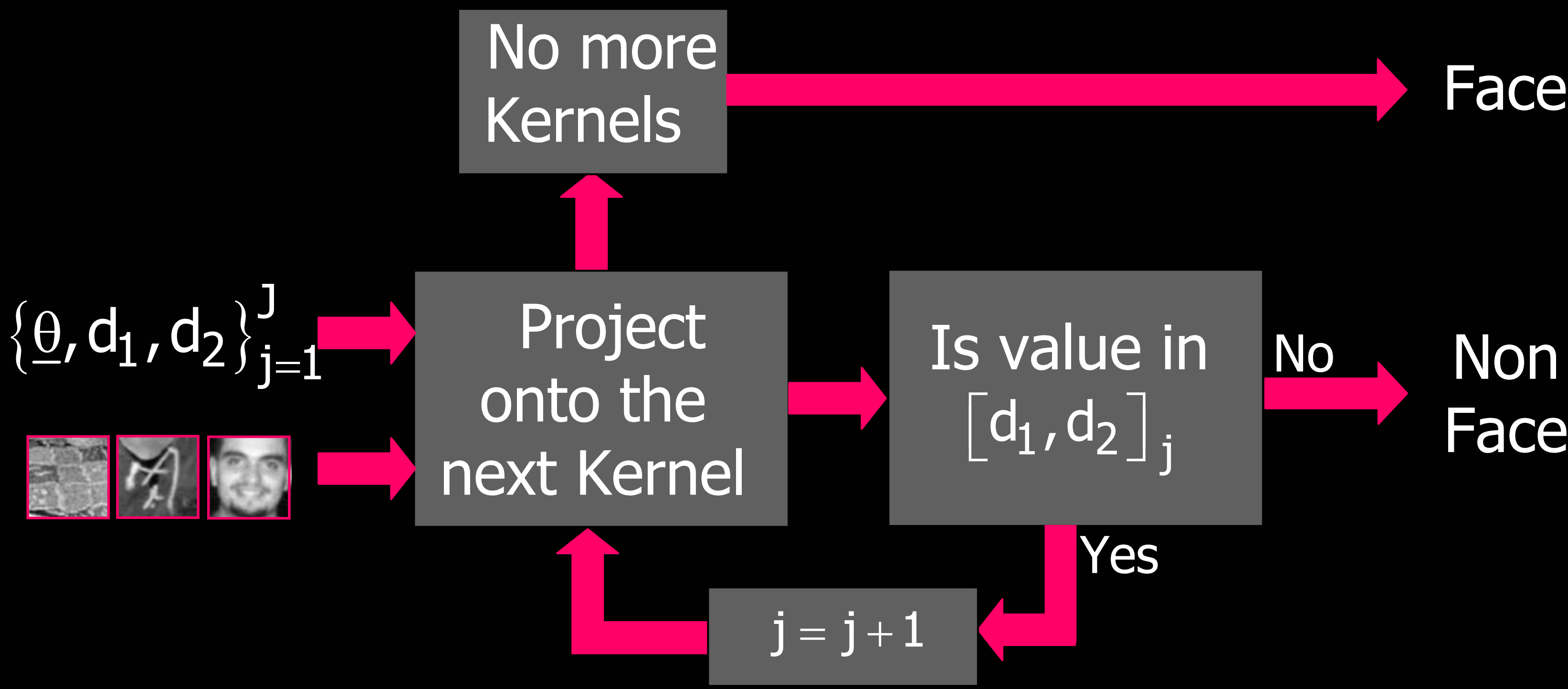
Minimize the inverse of the two expressions
(the inverse represent the proximity)

$$\text{Min} \ \frac{r_x^2}{(m_x - m_y)^2 + r_x^2 + r_y^2} + \frac{r_y^2}{(m_x - m_y)^2 + r_x^2 + r_y^2} = \text{Min} \ \frac{r_x^2 + r_y^2}{(m_x - m_y)^2}$$

back

4. Algorithm Structure




No more Kernels
Face
$\{\underline{\theta}, d_1, d_2\}_{j=1}^{J}$
Project onto the next Kernel
Is value in $[d_1, d_2]_j$
No
Non Face
Yes
$j = j + 1$

5. Counting Convolutions

- Assume that the first kernel rejection is $0<\alpha<1$ (i.e. α of the incoming blocks are rejected).
- Assume also that the other stages rejection rate is 0.5.
- Then, the number of overall convolutions per pixel is given by

$$\alpha \cdot 1 + (1-\alpha)\sum_{k=2}^{\infty} k \cdot 0.5^{k-1} = 3 - 2\alpha = \begin{cases} \sim 1 & \alpha = 0.99 \\ 1.2 & \alpha = 0.9 \\ 1.8 & \alpha = 0.6 \end{cases}$$

back

6. Using Color

Several options:

- Trivial approach – use the same algorithm with blocks of L-by-L by 3.
- Exploit color redundancy – work in HSV space with decimated versions of the Hue and the Saturation layers.
- Rejection approach – Design a (possibly non-spatial) color-based simple classifier and use it as the first stage rejection.

back

7. 2D-Rotated Faces

Input block → **Pose Estimation and Alignment** → **Frontal & Vertical Face Detector** → Face/ Non-Face

Remarks:

1. A set of rotated kernels can be used instead of actually rotating the input block
2. Estimating the pose can be done with a relatively simple system (few convolutions).

back

8. 3D-Rotated Faces

A possible solution:

1. Cluster the face-set to same-view angle faces and design a Final classifier for each group using the rejection approach
2. Apply a pre-classifier for fast rejection at the beginning of the process.
3. Apply a mid-classifier to map to the appropriate cluster with the suitable angle

back

9. Faces vs. Targets

- Treating other targets can be done using the same concepts of
  - Treatment of scale and location
  - Building and training sets
  - Designing a rejection based approach (e.g. MRC)
  - Boosting the resulting classifier
- The specific characteristics of the target in mind could be exploited to fine-tune and improve the above general tools.

back

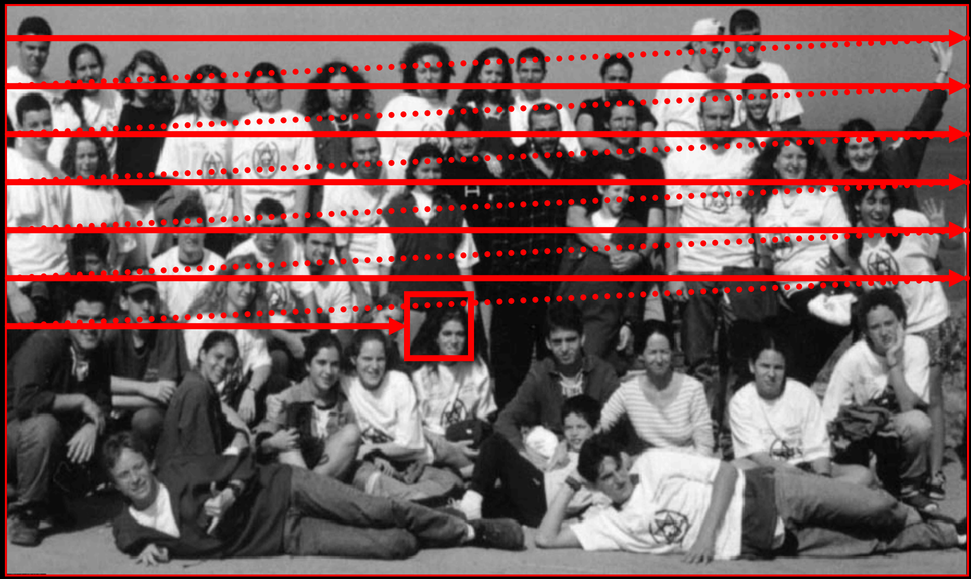
10. Further Improvements

- Pre-processing – linear kind does not cost
- Regularization – compensating for shortage in examples
- Boosted training – enrich the non-face group by finding false-alarms and train on those again
- Boosted classifier – Use the sequence of weak-classifier outputs and apply yet another classifier on them – use ada-boosting or simple additional linear classifier
- Constrained linear classifiers for simpler classifier
- Can apply kernel methods to extend to non-linear version

back

1. Algorithm Complexity

Searching targets in a given scale, for a 1000 by 1000 pixels image, the classifier is applied 1e6 times (even if no scale is involved!!)

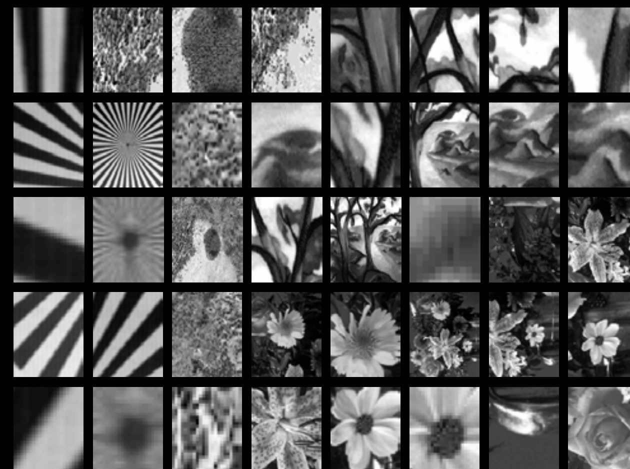


(Q1) Choosing the parametric form:
keep in mind that the algorithm's complexity is governed by the classifier complexity.

Interesting idea: apply spatially varying classifier!?

2. Training by Examples

$\left\{\underline{X}_k\right\}_{k=1}^{N_X}$ $\left\{\underline{Y}_k\right\}_{k=1}^{N_Y >> N_X}$



(Q2) Finding Suitable Parameters:

$$\forall 1 \le k \le N_X,\ \ C\left\{\underline{X}_k, \underline{\theta}\right\} = +1$$

$$\forall 1 \le k \le N_Y,\ \ C\left\{\underline{Y}_k, \underline{\theta}\right\} = -1$$

While allowing outliers for better generalization behavior

3. Exploiting Our Knowledge

If we know that indeed:

- Volume{*Target*} << Volume{*Clutter*},
- Prob{*Target*} << Prob{*Clutter*}, and
- The *Target* class is nearly convex,

We would like to obtain:

- Simpler parametric form for the classifier,
- Simpler/faster training algorithm,
- Faster classifier,
- A classifier with spatially dependent complexity, and
- More accurate classifier.