# Face Detection with Colors

Three Challenges

# Outlines

- Project Groups
- Performance Evaluation
- Face Detection using Colors
  - Challenge 1: Detecting skin pixels with colors
  - challenge 2: Detecting Faces with skin color
  - challenge 3: Face localization

# Outlines

- Project Groups
- Performance Evaluation
- Face Detection using Colors
  - Challenge 1: Detecting skin pixels with colors
  - challenge 2: Detecting Faces with skin color
  - challenge 3: Face localization

# Performance Evaluation

## *0. Introduction*

# Performance Evaluation of Detectors

A pattern detector is a classifier with K=2.

Class k=1: The target pattern, also known as P or positive

Class k=2: Everything else, also known as N or negative.

# Performance Evaluation of Detectors

Assume that we have M training samples,

- **X** are the training samples
- **Y(x)** a function that tells if x is in the target class P or N.

The pattern detector is learned as a detection function ***g(X)*** followed by a decision rule, ***d(g(X))***.

For example, the decision rule can be :

$$\text{if} \quad g\left(\vec{X}\right) + B \geq 0.5 \quad \text{then} \quad \text{P else N}$$

# Performance Evaluation of Detectors

For example, the decision rule can be :

$$\text{if} \quad g\left(\vec{X}\right) + B \geq 0.5 \quad \text{then} \quad \text{P else N}$$

Therefore, the detection function R(X) can rewritten as:

$$R(\vec{X}) = d(g(\vec{X}) + B) = \begin{cases} P & \text{if } g(\vec{X}) + B \geq 0.5 \\ N & \text{if } g(\vec{X}) + B < 0.5 \end{cases}$$

# Performance Evaluation of Detectors

In order to evaluate the detector, we need ground truth information.

$$y_m = \begin{cases} P & \vec{X}_m \in \text{Target - Class} \\ \text{N} & \text{otherwise} \end{cases}$$

The classification can be **TRUE**, or **FALSE**

$$\text{if } R(\vec{X}_m) = y_m \text{ then T else F}$$

# Performance Evaluation of Detectors

This results in one the following cases:

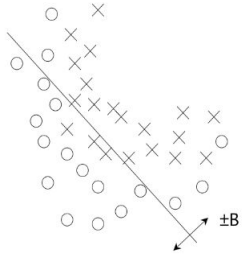
$$R(\vec{X}_m) = y_m \text{ AND } R(\vec{X}_m) = P \text{ is a TRUE POSITIVE or TP}$$
$$R(\vec{X}_m) \neq y_m \text{ AND } R(\vec{X}_m) = P \text{ is a FALSE POSITIVE or FP}$$
$$R(\vec{X}_m) \neq y_m \text{ AND } R(\vec{X}_m) = N \text{ is a FALSE NEGATIVE or FN}$$
$$R(\vec{X}_m) = y_m \text{ AND } R(\vec{X}_m) = N \text{ is a TRUE NEGATIVE or TN}$$

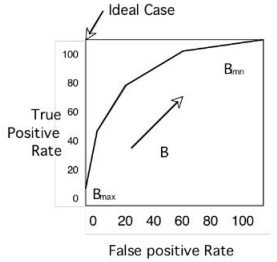
# Performance Evaluation

## *1. ROC curve*

# Receiver Operating Characteristic (ROC) Curve

- Two-class classifiers have long been used for signal detection problems in communications and have been used to demonstrate optimality for signal detection methods.

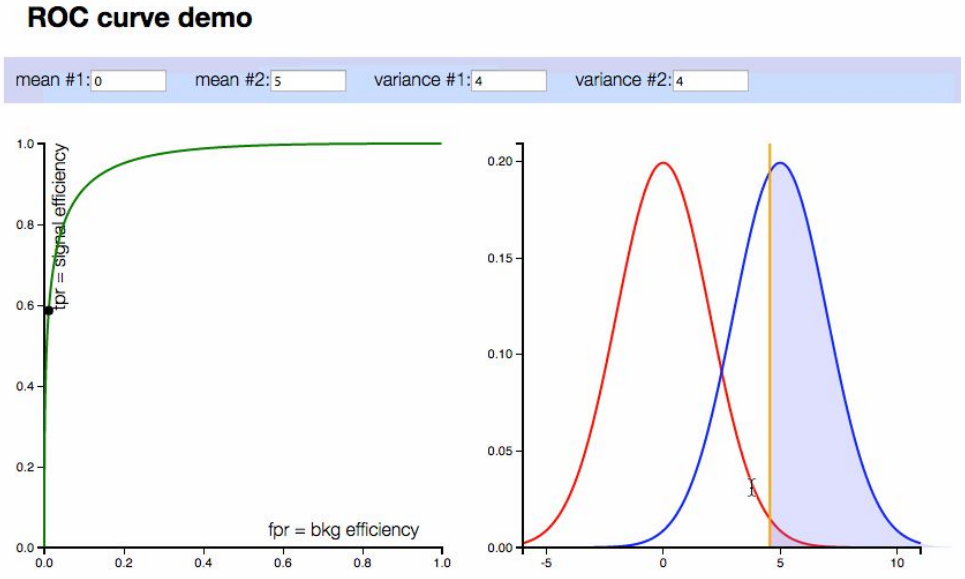



- The quality metric that is used is the Receiver Operating Characteristic (ROC) curve.

- This curve can be used to describe or compare any method for signal or pattern detection.

# Receiver Operating Characteristic (ROC) Curve



# Receiver Operating Characteristic (ROC) Curve

The ROC curve is generated as following:

1. adding a variable Bias term to a discriminant function. $R(\vec{X}) = d(g(\vec{X}) + B)$
2. plotting the rate of True Positive detection (TPR) vs False Positive detection (FPR).

$$TPR = \frac{\#TP}{\#P} = \frac{\#TP}{\#TP + \#FN} \qquad FPR = \frac{\#FP}{\#N} = \frac{\#FP}{\#FP + \#TN}$$

As the bias term, B, is swept through a range of values, it changes the ratio of true positive detection to false positives.

# Receiver Operating Characteristic (ROC) Curve

- The choice of B:
  For a ratio of histograms, $g(X_m)$ is a probability ranging from 0 to 1.
  The bias term, B, can act as an adjustable gain that sets the sensitivity of the detector.

B can range from less than $-0.5$ to more than $+0.5$.
When $B \leq -0.5$ all detections will be Negative.
When $B > +0.5$ all detections will be Positive.
Between $-0.5$ and $+0.5$ $R(\vec{X})$ will give a mix of TP, TN, FP and FN.

# Performance Evaluation

## *2. Precision and Recall*

# Precision and Recall

- **Confusion Matrix,**

| | | $y_m = R(\vec{X}_m)$ | |
|---|---|---|---|
| $d(g(\vec{X}_m)+B > 0.5)$ | | T | F |
| | P | True Positive (TP) | False Positive (FP) |
| | N | True Negative (TN) | False Negative (FN) |

# Precision and Recall

- **Precision,** also called Positive Predictive Value (PPV), is the fraction of retrieved instances that are relevant to the problem.

$$PP = \frac{TP}{TP + FP}$$

A perfect precision score (PPV=1.0) means that every result retrieved by a search was relevant, but says nothing about whether all relevant documents were retrieved.

# Precision and Recall

- **Recall,** also known as sensitivity (S), hit rate, and True Positive Rate (TPR) is the fraction of relevant instances that are retrieved.

$$S = TPR = \frac{TP}{T} = \frac{TP}{TP + FN}$$

A perfect recall score (TPR=1.0) means that all relevant documents were retrieved by the search, but says nothing about how many irrelevant documents were also retrieved.

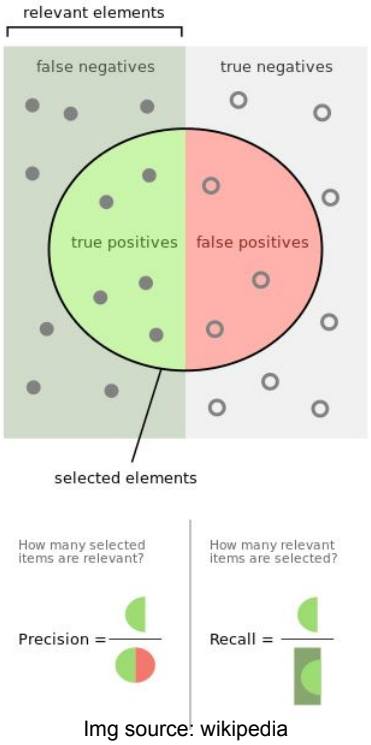
# Precision and Recall

Both precision and recall are therefore based on an understanding and measure of relevance. In our case, “relevance” corresponds to “True”.

**Precision** answers the question “**How many** of the **Positive Elements** are **True** ?”

**Recall** answers the question “**How many** of the **True elements** are **Positive**”?

In many domains, there is an inverse relationship between precision and recall. It is possible to increase one at the cost of reducing the other.




Img source: wikipedia

# Performance Evaluation

## *3. F-Measure*

# F- Measure

The F-measures combine precision and recall into a single value. It has a general form:

$$F_\beta = \frac{(1+\beta^2)\cdot(\mathrm{precision}\cdot\mathrm{recall})}{(\beta^2\cdot\mathrm{precision}+\mathrm{recall})}$$

**F1-score**, is the harmonic mean of precision and sensitivity.

$$F_1 = \left(\frac{\mathrm{recall}^{-1}+\mathrm{precision}^{-1}}{2}\right)^{-1} = 2\cdot\frac{\mathrm{precision}\cdot\mathrm{recall}}{\mathrm{precision}+\mathrm{recall}}$$

**$F_2$ measure**, which weighs recall *higher* than precision (by placing more emphasis on false negatives),
**$F_{0.5}$ measure**, which weighs recall *lower* than precision (by attenuating the influence of false negatives).

# Performance Evaluation

## *4. Accuracy*

# Accuracy

**Accuracy,** is the fraction of test cases that are correctly classified (T).

$$ACC = \frac{T}{M} = \frac{TP + TN}{M}$$

where M is the quantity of test data.

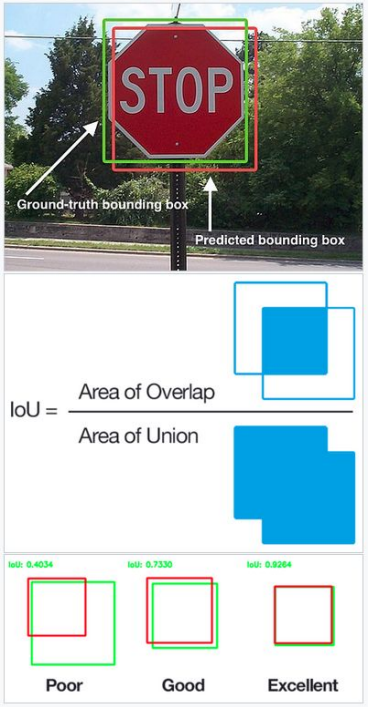
# Performance Evaluation

## 5. IoU

# Intersection Over Union (IoU)

**IoU,** also known as **Jaccard index** is measures similarity between finite test sets, is defined as the size of the intersection divided by the size of the union of the test sets.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$



# Performance Evaluation

## 6. AP

# Average Precision (AP)

For detectors that return a ranked sequence of predictions,

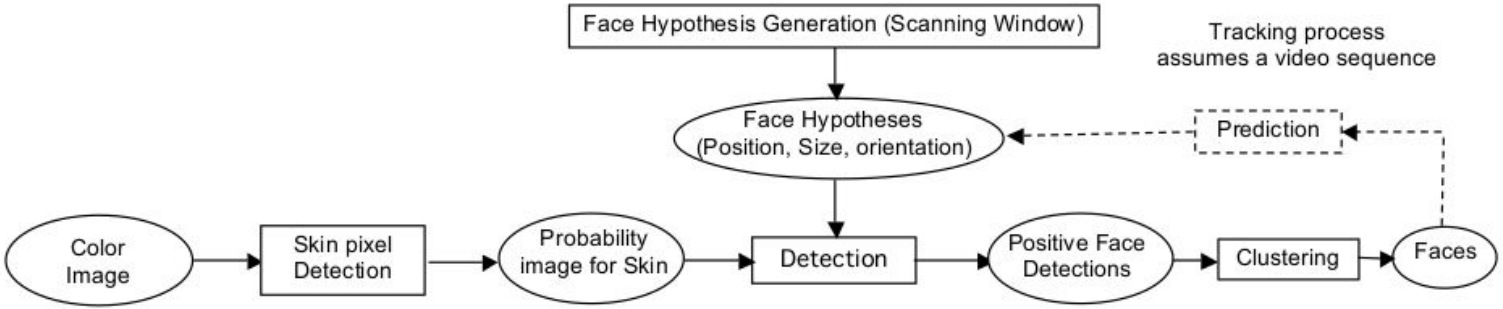
- Compute a precision and recall at every position in the ranked sequence of predictions,
- Plot a precision-recall curve, plotting precision p (r) as a function of recall r.
- Average precision computes the average value of p (r) over the interval from r = 0 to r = 1.

$$\text{AveP} = \int_0^1 p(r)dr$$

# Face Detection using skin color

## *0. Intro*

# Face Detection using Skin Color



# Face Detection using Skin Color

**Algorithm outline:**

1. Compute probability of skin at each pixel.
2. Test for faces at possible positions and sizes (scanning Window).
3. Cluster adjacent detections.
4. Estimate precise face position, size and orientation from clusters of detections.

# Face Detection using skin color

## *1. Detecting skin pixels*

# Baseline: Ratio of RGB histograms

Transform a color image (RGB) into an image of probabilities of skin P(i,j)

Assume a color image : $X(i,j) = \begin{pmatrix} R \\ G \\ B \end{pmatrix}(i,j)$

The algorithm will use a lookup table to convert color to probability.

$$P(i,j) \leftarrow L(\vec{X}(i,j))$$

# Baseline: Ratio of RGB histograms

1) Suppose that we have N color images of size CxR pixels, $X_n$ (i,j). This gives a total of M = C × R × N pixels.
2) Suppose that we have a ground truth function Y($X_m$) that tells us whether each pixel is target or not target. A subset of $M_T$ pixels that belong to a target class, T.
3) Compute 2 histograms as following:

Histogram of **All** pixel colors $\forall_m h(\vec{X}_m) = h(\vec{X}_m) + 1$

Histogram of **Skin** pixel colors $\forall_m y(\vec{X}_m) = P : \; h_T(\vec{X}_m) = h_T(\vec{X}_m) + 1 \; ; \; M_T \leftarrow M_T + 1$

# Baseline: Ratio of RGB histograms

For a color vector, X = (R, G, B) we have two probabilities: $P(\vec{X}) = \frac{1}{M} h(\vec{X})$ and $P(\vec{X} \mid T) = \frac{1}{M_T} h_T(\vec{X})$

Using Bayes Rule: $P(Skin \mid \vec{X}) = \frac{P(\vec{X} \mid \text{Skin}) P(\text{Skin})}{P(\vec{X})}$

P(Skin) is the probability that a pixel belongs to the target class. $P(Skin) = \frac{M_T}{M}$

Our lookup table is:

$$P(Skin \mid \vec{X}) = \frac{P(\vec{X} \mid Skin) P(Skin)}{P(\vec{X})} = \frac{\frac{1}{M_T} h_t(\vec{X}) \cdot \frac{M_t}{M}}{\frac{1}{M} h(\vec{X})} = \frac{h_T(\vec{X})}{h(\vec{X})}$$

# Baseline: Ratio of RGB histograms

How large are those tables? Each has $V = 2^8 \cdot 2^8 \cdot 2^8 = 2^{24}$ cells.

**Parameter to adjust!**

- Different color quantizations,
  - Encoding each color histogram with 3 bits instead of 8 bits, results in smaller tables.
  - How many bits are sufficient for skin color detection task?
  - How does it affect recognition rate?

# Variation: Ratio of RGB histograms

**Detection with Chrominance (normalized color),**

- Luminance captures local surface orientation (3D shape)
- Chrominance is a signature for object pigment (identity). The color of pigment for any individual is generally constant.
- Luminance can change with pigment density (eg. Lips), and skin surface orientation, but chrominance will remain ***invariant***.

# Variation: Ratio of RGB histograms

**Detection with Chrominance (normalized color),**

- A popular color space for skin detection:

$$\begin{pmatrix} L \\ c_1 \\ c_2 \end{pmatrix} \Leftarrow \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

Luminance: L= R+G+B

Chrominance : $r = c_1 = \frac{R}{R+G+B}$ $g = c_2 = \frac{G}{R+G+B}$

$$r = trunc\left((Q-1)\cdot\frac{R}{R+G+B}\right) \qquad g = trunc\left((Q-1)\cdot\frac{G}{R+G+B}\right)$$

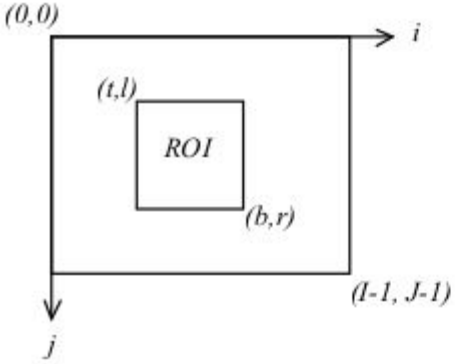
# Face Detection using skin color

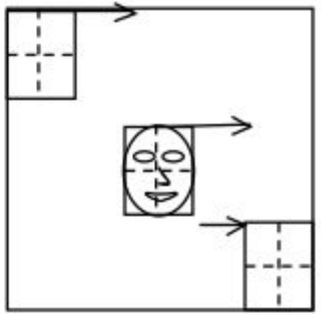
## *2. Sliding window detection*

# Baseline: Sliding window detector

Each window is face hypothesis defined as: (t, l, b, r)

t - "top" - first row of the ROI.
l - "left" - first column of the ROI.
b - "bottom" - last row of the ROI
r - "right" - last column of the ROI.



# Baseline: Sliding window detector

Sliding window parameters:

- Window size = (window height, window width)
- Sliding step = (vertical skip, horizontal skip)

For each window, we compute the likelihood of a face at a position (ci, cj) of size (w, h):

$$g(\vec{X}) = \frac{1}{w \cdot h} \sum_{i=l}^{r} \sum_{j=t}^{b} P(i,j)$$

# Variation: Sliding window using a Gaussian mask.

In machine vision, fixation serves to reduce computational load and reduce errors by focusing processing on parts of the image that are most likely to contain information.

A gaussian mask is defined as:

$$G(i,j;\vec{\mu},\Sigma) = \frac{1}{2\pi \det(\Sigma)^{1/2}} e^{-\frac{1}{2}\left(\vec{P}-\vec{\mu}\right)^T \Sigma^{-1}\left(\vec{P}-\vec{\mu}\right)}$$

P= image positions, $\mathcal{M}$ is the center position of the gaussian mask, $\det(\Sigma) = \sigma_i^2\sigma_j^2 - \sigma_{ij}^2$

# Variation: Sliding window using a Gaussian mask.

Based on Gaussian mask we can define a face hypothesis as $\vec{X} = \begin{pmatrix} \mu_i \\ \mu_j \\ \sigma_i^2 \\ \sigma_j^2 \\ \sigma_{ij} \end{pmatrix}$

We can define the ROI as a 3σ bounding box:

$$t = c_j - 3\sigma_j, \qquad b = c_j + 3\sigma_j, \qquad l = c_i - 3\sigma_i, \qquad r = c_i + 3\sigma_i$$

We compute the likelihood of a face hypothesis as:

$$g(\vec{X}) = \sum_{i=l}^{r} \sum_{j=t}^{b} P(i,j) G(i,j;\vec{X})$$

# Parameters to be evaluated.

For a simple scanning window, these include:

- Width and height of ROI
- Range of positions
- Step size for scanning windows
- The percentage of overlap with the ground truth that is considered a TRUE detection.

For a Gaussian detection window, parameters also include

- The width and height of the ROI compared to the standard deviations of the principal axis.
- The range and step sizes for orientation of the Gaussian window.

# Face Detection using skin color

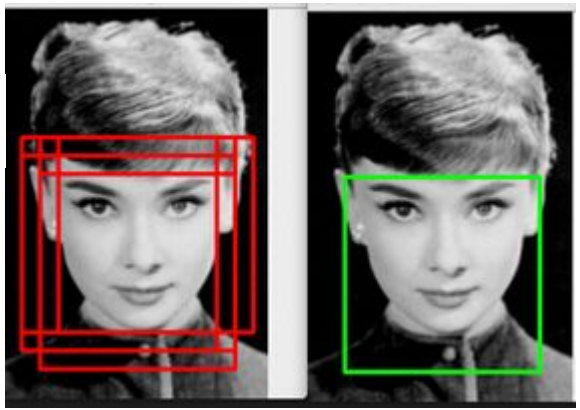
## *3. Face Localization*

# Baseline: Localization by non-maximum suppression

In order to suppress non-maximal detections, the easiest method is to build a list of detections hypotheses, { X } over the desired range of positions, scales, orientations and any other parameters, and then filter this list to remove any hypothesis for which the discriminant not a local maximum.

$$\forall \vec{X}_i, \vec{X}_j \in \{\vec{X}\} : \text{IF } Dist(\vec{X}_i, \vec{X}_j) < R \text{ AND } g(\vec{X}_i) < g(\vec{X}_j) \text{ THEN } \{\vec{X}\} \leftarrow \{\vec{X}\} - \vec{X}_i$$

Distance that includes different parameters:
ex. Mahalanobis distance



# Variation: Localization by Robust Estimation

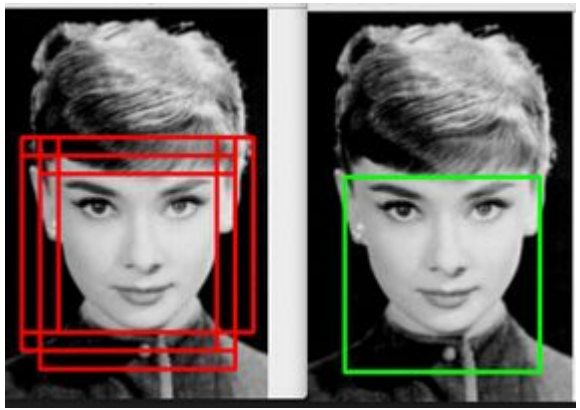
- Suppose that we have a set of N detections: {Xn} and for each detection we have a discriminant g(Xn).

Mass of detection (zeroth moment of detection set):

$$M = \sum_{n=1}^{N} g(\vec{X}_n)$$

The expected value (First moment of detection set):

$$E\{\vec{X}\} = \frac{1}{M} \sum_{n=1}^{N} g(\vec{X}_n) \cdot \vec{X}_n$$



# Variation: Localization by Robust Estimation

- This is the first step of Expectation Maximization Algorithm for multiple parameter estimations.

**Parameters to test,**

- Number of Faces in the image
- Range of positions, size and orientations to test.
- Distance to use for non-maximal suppression.
- Size of the region used to cluster faces detections.