OPTOMATICA

THE POWER TO PERFECT

Face Recognition:
Tech. that knows you!

“All, all are gone, the old familiar faces.”

— Charles Lamb

OPTOMATICA
THE POWER TO PERFECT

Face Recognition

The ability to recognize a person’s face in a digital image.

Early Days

- First real system was created in 1964 by Woody Bledsoe, Helen Chan Wolf and Charles Bisson.
- Operator had to manually measure and locate key points on the face.
- System could only filter down a list pictures to a small set of possible matches.

GPUs and NNs revolt

- Systems using increasingly complex statistical models were built from the 1970s through the 2000s.

- The availability of GPUs and the advent of deep learning through the mid 2010s.

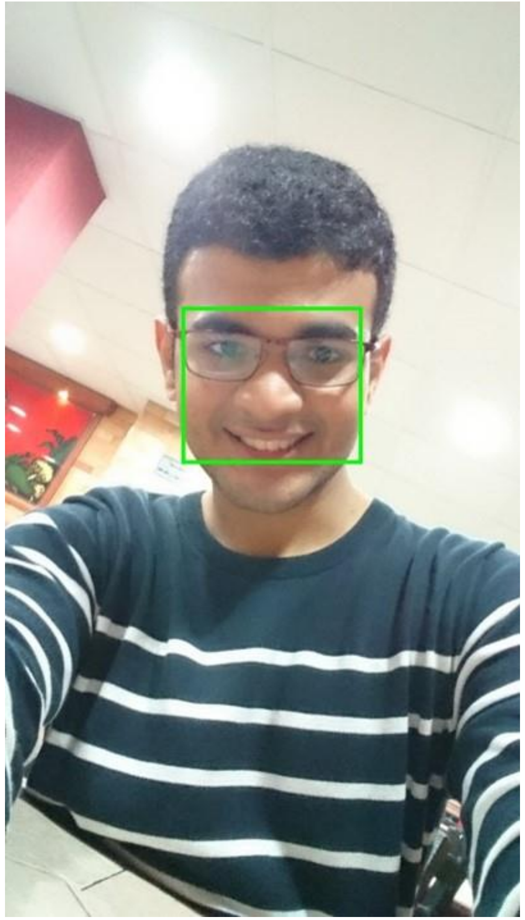
Wild Apps

- Key research papers from Google and Facebook in 2014 and 2015 demonstrated how to reach near human-level performance on face recognition.

- Available to almost anyone for all kinds of products.

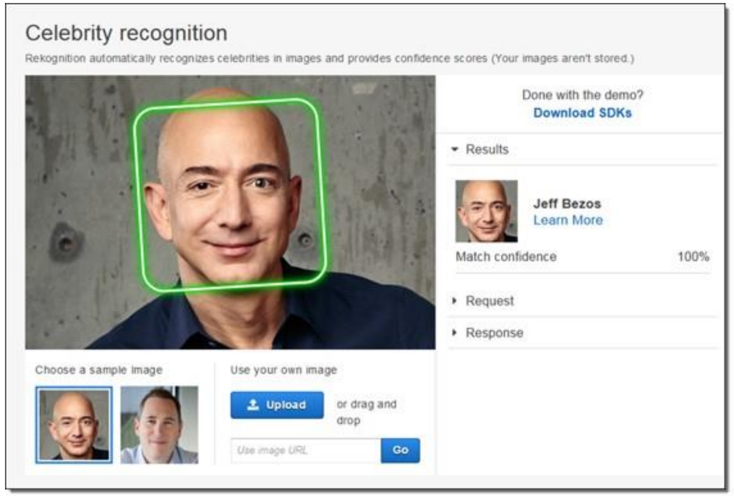
Uses

- Identity Verification
- Tagging Photos
- Surveillance
- Advertisement

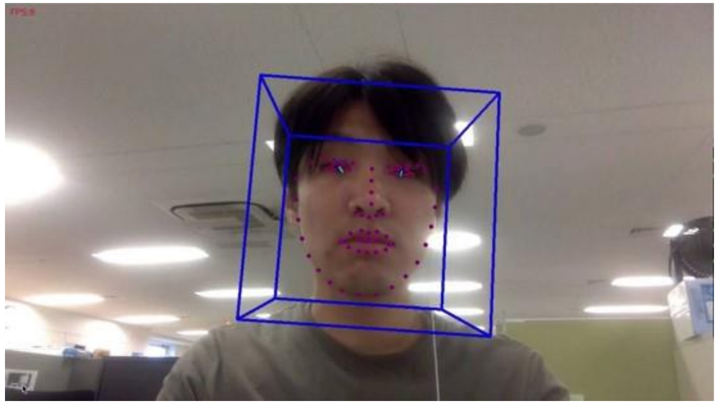


Tools

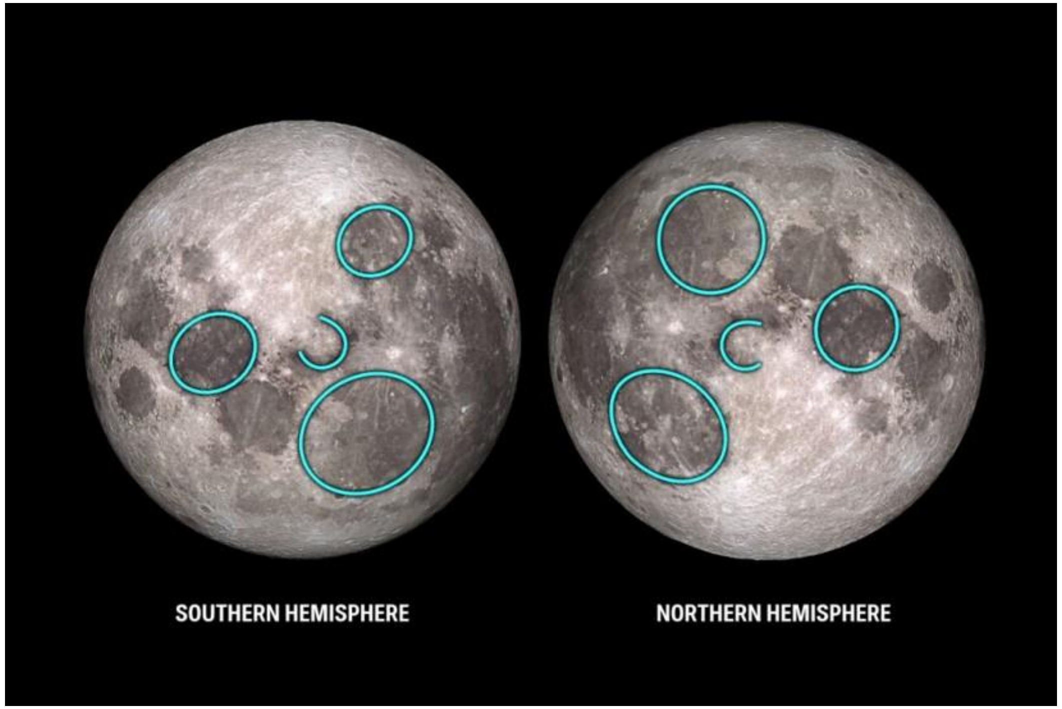
- Commercial Services
  - Amazon Rekognition
  - Microsoft Azure Face
  - Google Cloud Vision
  - IBM Watson Visual Recognition



- Open-Source
  - OpenFace by Brandon Amos at Carnegie Mellon.
  - Dlib by Davis King.



Pareidolia



Pareidolia

OPTOMATICA
THE POWER TO PERFECT

Multi-Step Pipeline

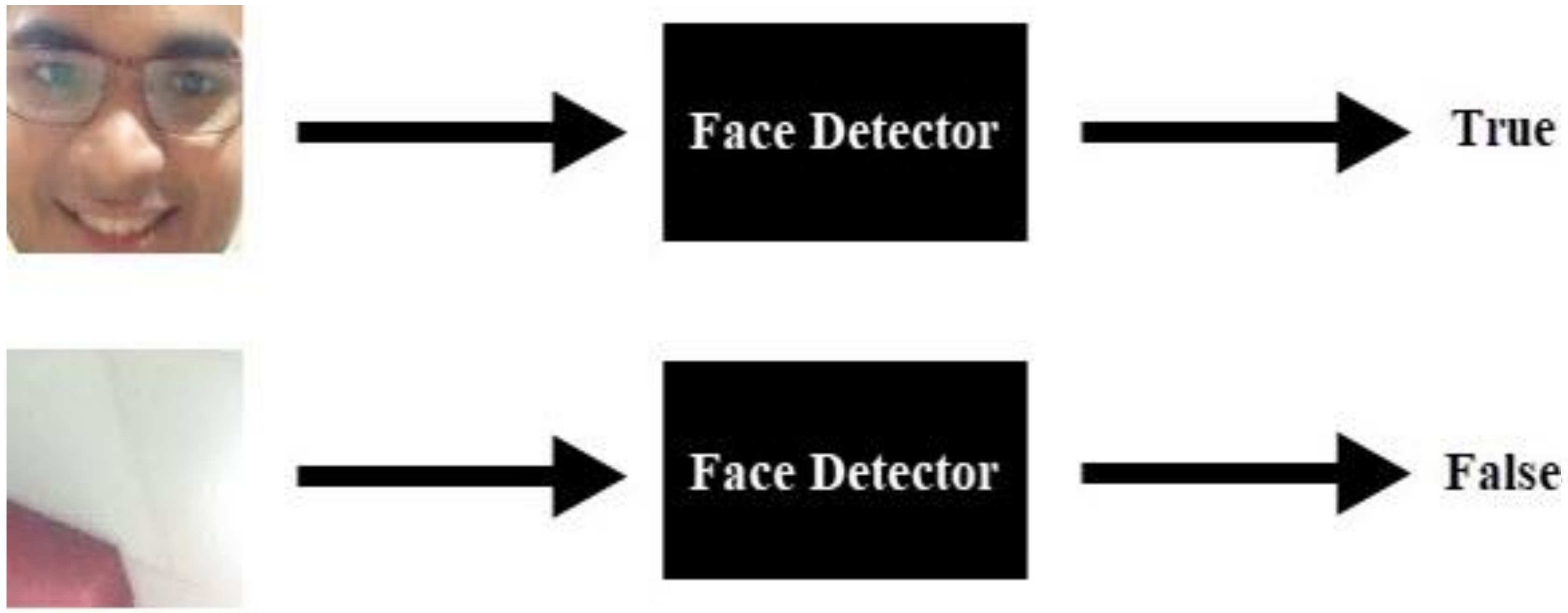
OPTOMATICA
THE POWER TO PERFECT

- Locate and extract faces from the image.
- Identify face landmarks.
- Align faces to match a pose template.
- Encode faces.
- Check the Euclidean distance between face encodings.

Face Detection

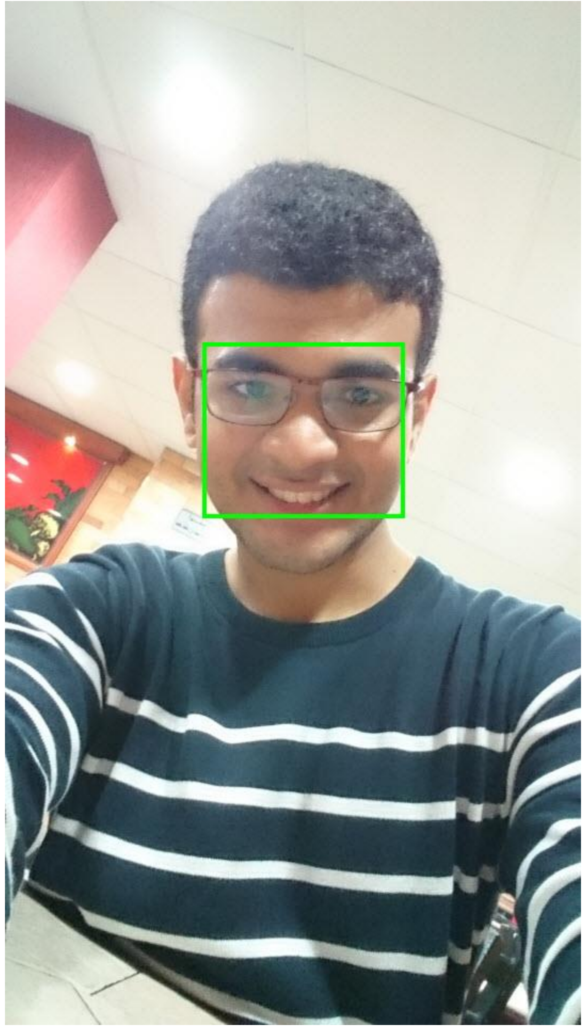
The ability to detect and locate human faces in an image.

Step 1: Build a face detector.



Step 2: Slide a window.

Step 3: Return location.



Face Classification

OPTOMATICA
THE POWER TO PERFECT

- Viola-Jones
- Histogram of Gradients or HOG
- Convolutional Neural Networks or CNNs

Viola-Jones

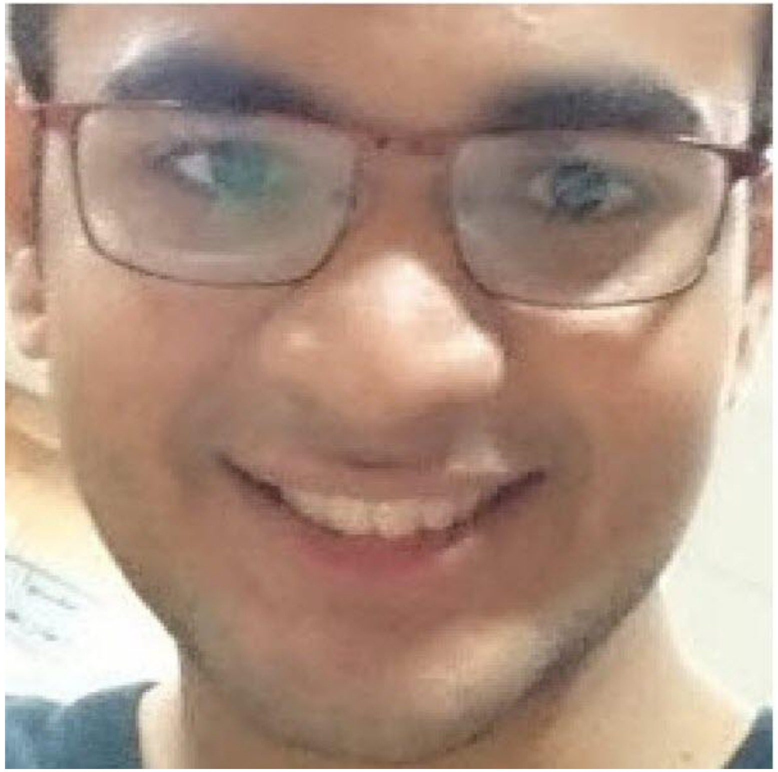
- Invented by Paul Viola and Michael Jones in the early 2000s.

- Very fast and great for low-powered devices.

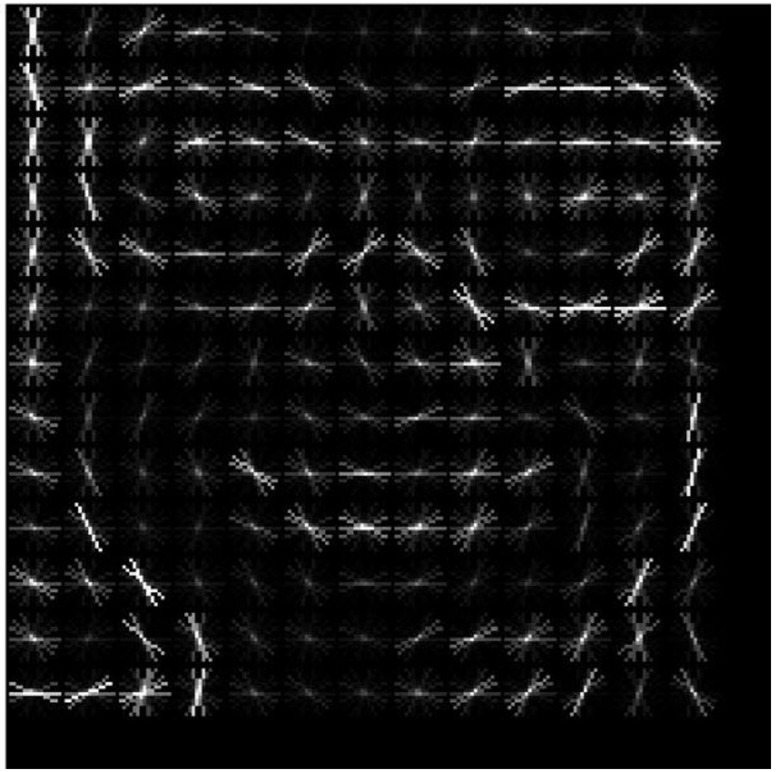
HOG: Conversion

OPTOMATICA
THE POWER TO PERFECT

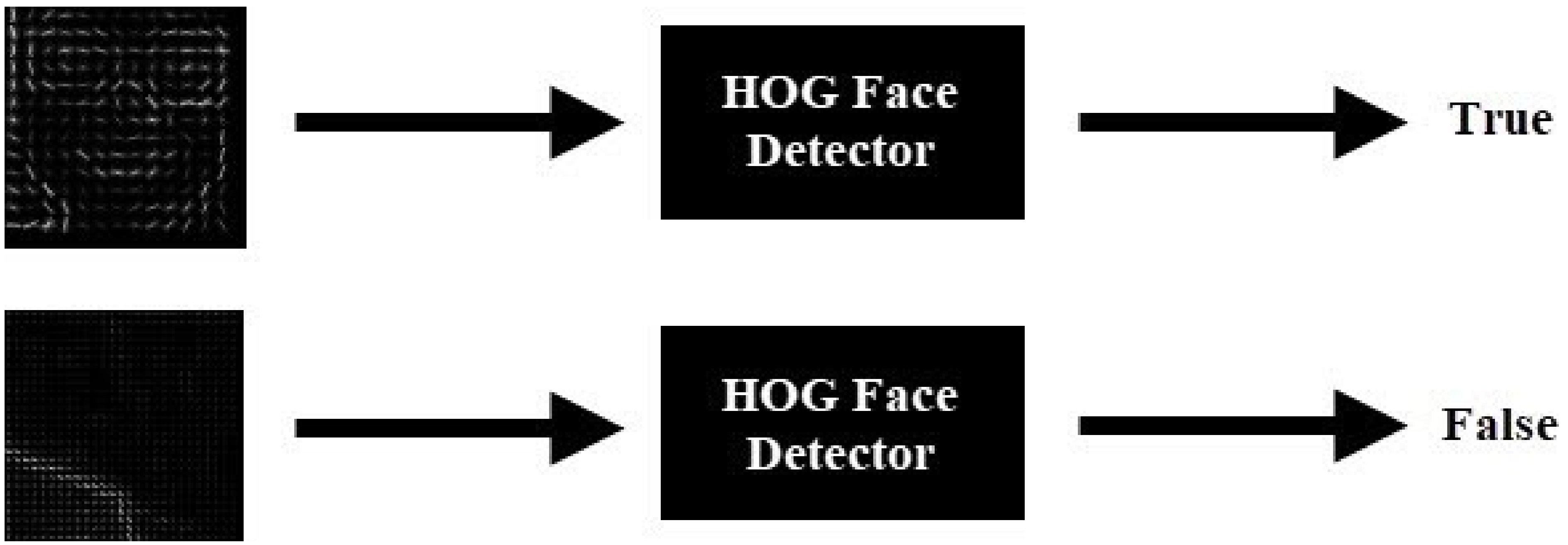
Input Image



HOG Representation

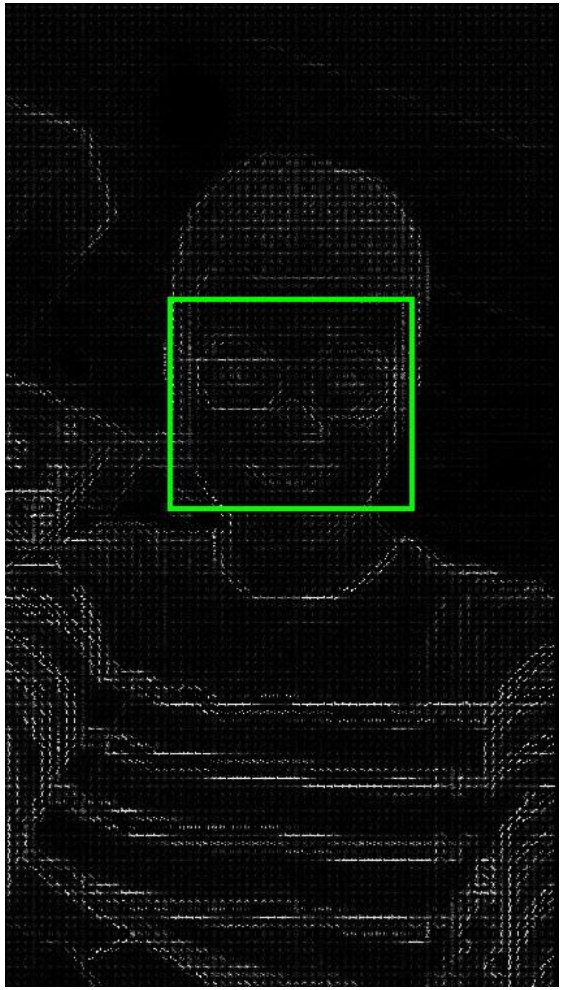


Step 1: Build a HOG face detector.



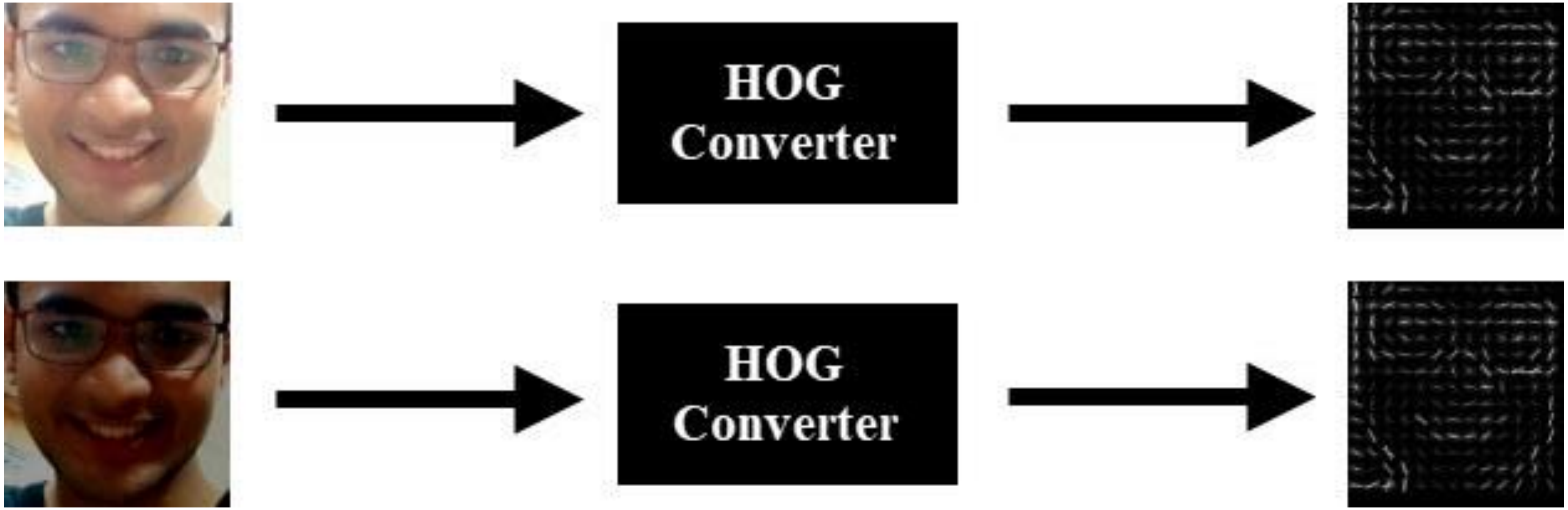
Step 2: Slide a window.

Step 3: Return location.



HOG: Advantages

- Compactness with enough detail to detect faces.
- Resistance to small changes in lighting and object’s shape.



CNNs

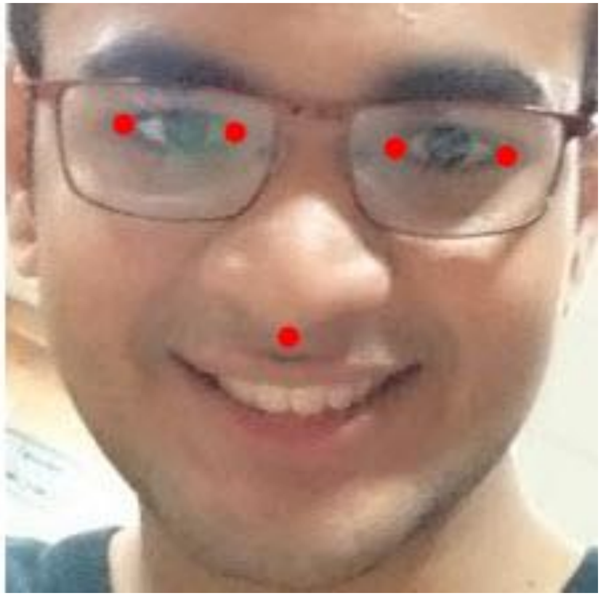
- Most accurate solution.
- Lots of training data.
- Slow unless backed by a GPU to offload processing.

Face Landmark Estimation

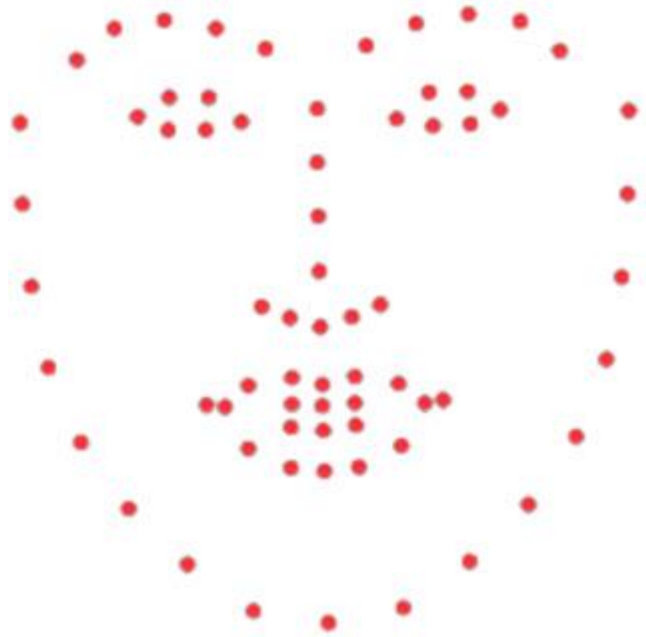
Locating key points on a face, like the center of the eyes and mouth.

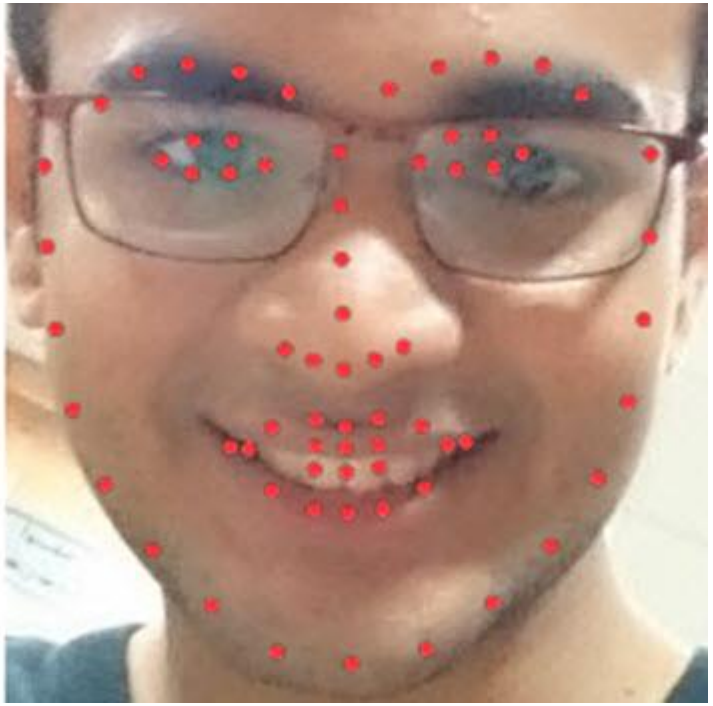
Face Landmark Estimation

Face Landmark Estimation

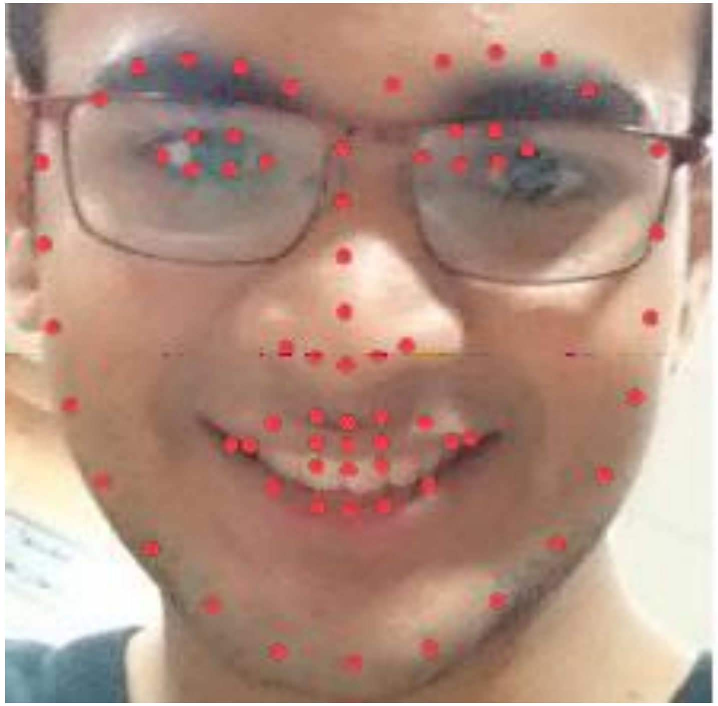


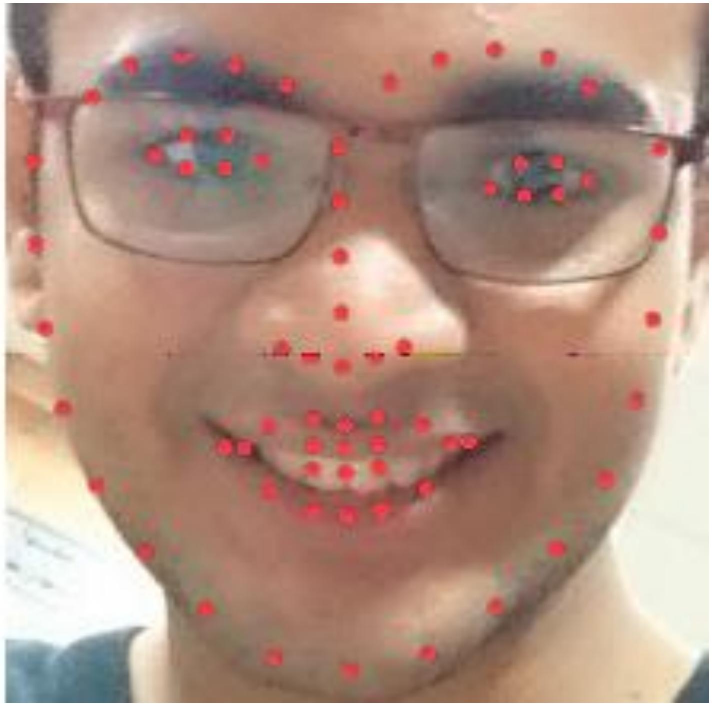
Trick 1: Assume all faces are similar.



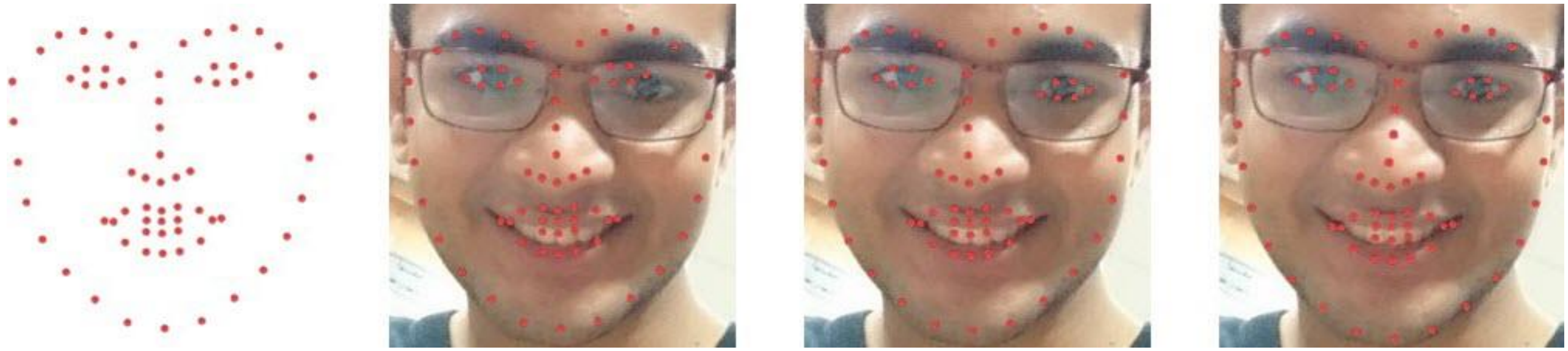


Trick 2: Limit movement of each point.





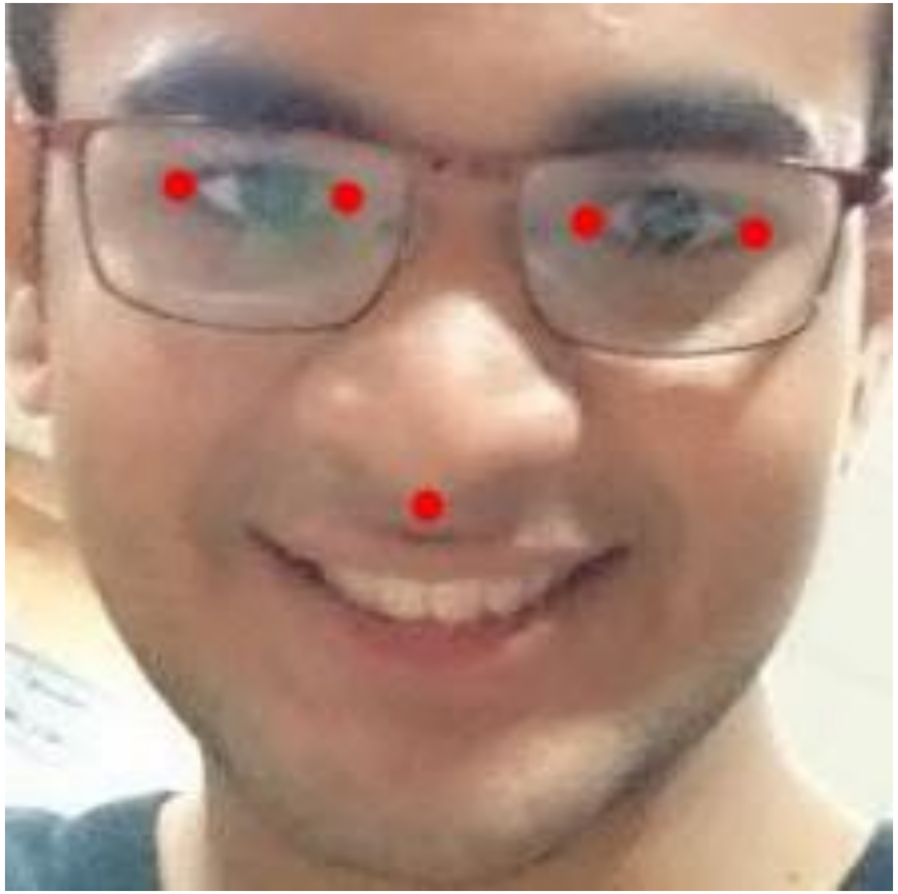
Trick 3: Fine-tune with multiple models.



Face Alignment

Adjusting a raw face image so that key facial features like the eyes, nose and mouth line up with a predefined template.

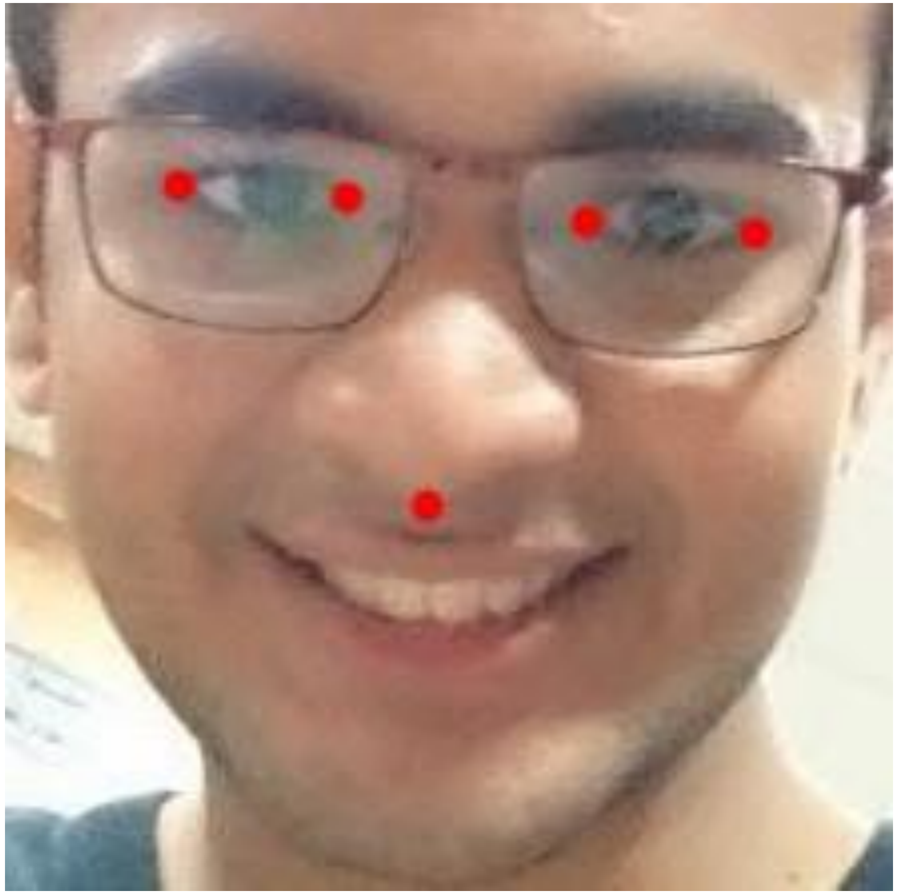
Step 1: Detect face landmarks.

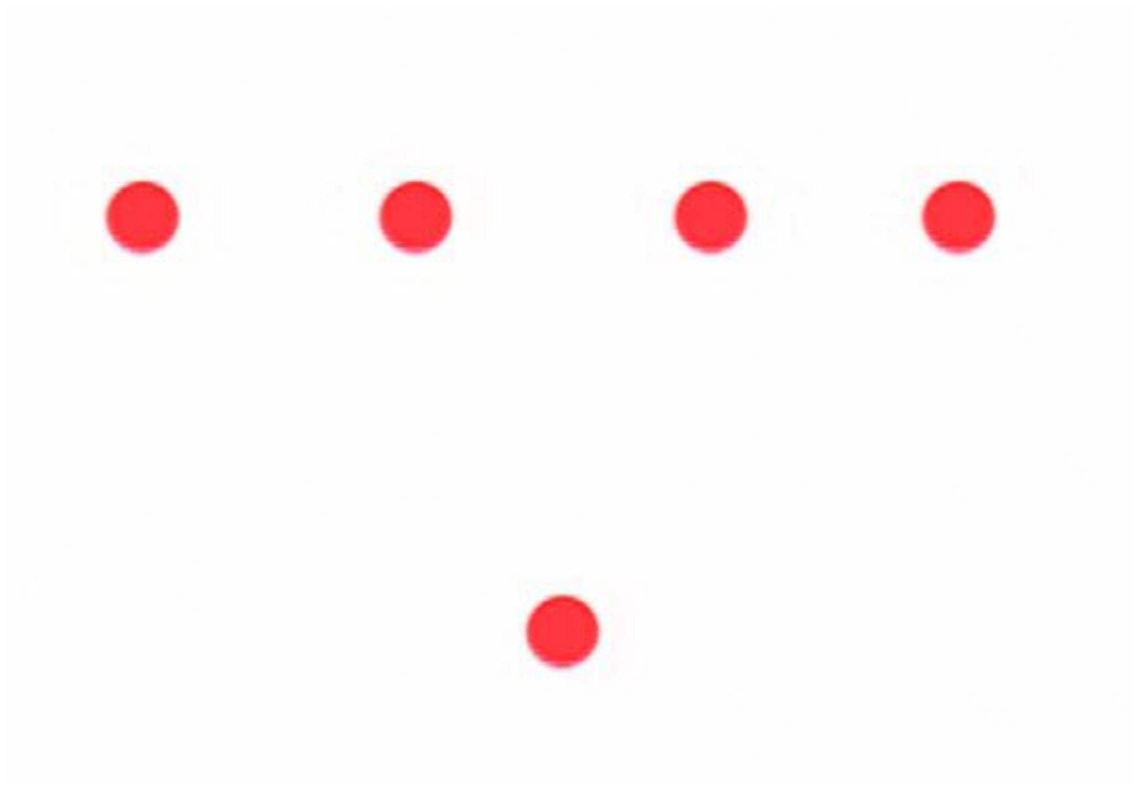


OPTOMATICA
THE POWER TO PERFECT

Step 2: Calculate the Affine Transform.

OPTOMATICA
THE POWER TO PERFECT



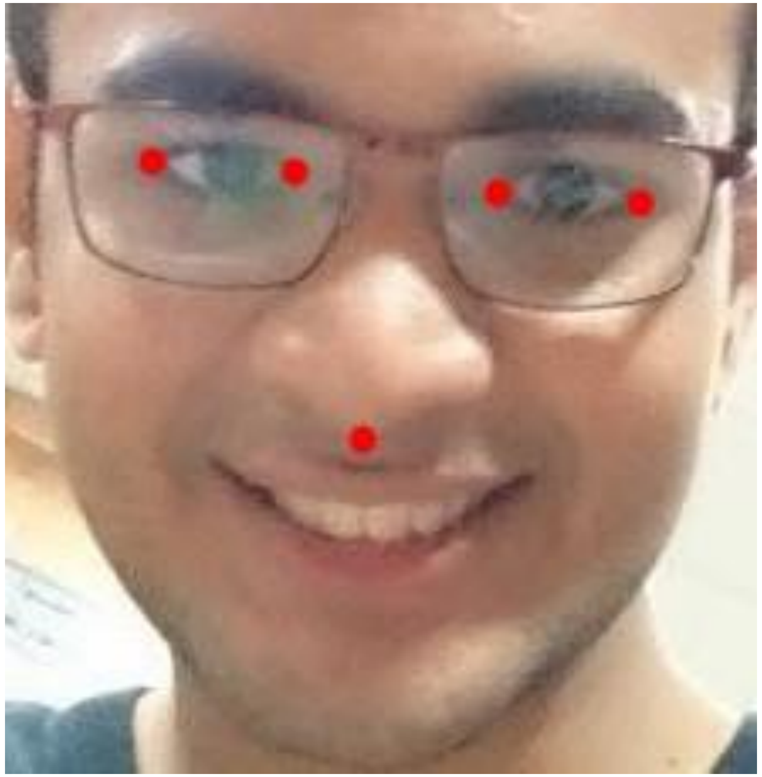


Affine Transformation

A linear mapping between sets of points where parallel lines remain parallel.

Step 2: Calculate the Affine Transform.

OPTOMATICA
THE POWER TO PERFECT



Identifying an Unknown Face

Unknown Face

Known Faces

Comparing Faces

OPTOMATICA
THE POWER TO PERFECT

Face Encoding

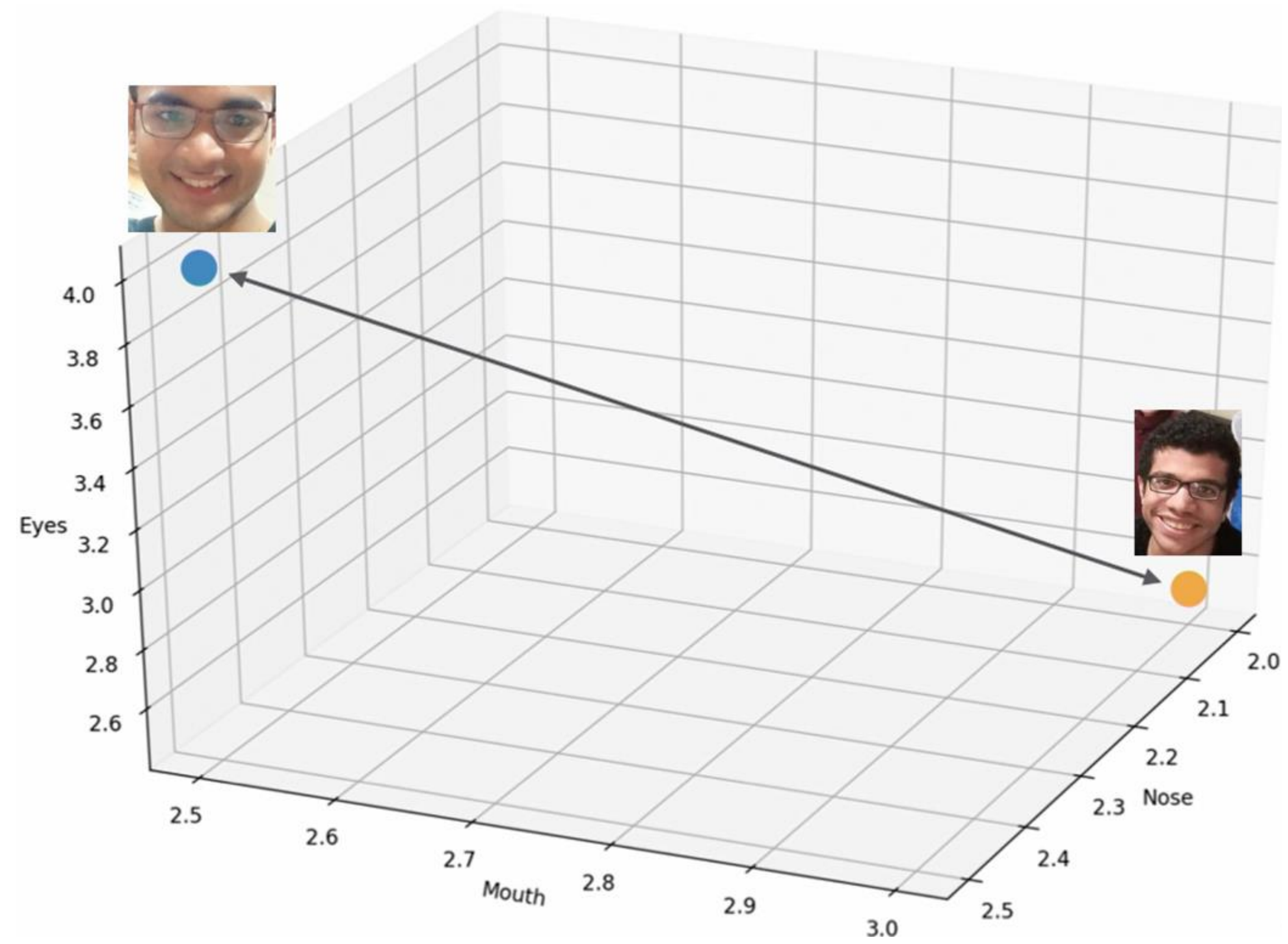
The process of taking a face image and turning it into a set of measurements.

Measuring Faces

OPTOMATICA
THE POWER TO PERFECT

	Left	Right
Nose	2.5	2
Mouth	2.5	3
Eyes	4	2.5

Reduced Faces to Points in 3D Space



Euclidean Distance

The distance between two points in space along a straight line.

Face Distance Threshold

- Maximum face distance at which the face is considered the same.
- If $distance(Left, Right) > 0.6$ then they're not a match.
- If $distance(Left, Right) \leq 0.6$ then they're a match.
- The lower the distance, the better the match.

Calculating Euclidean Distance

$$distance(a, b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + (a_3 - b_3)^2}$$

$$distance(Left, Right) = \sqrt{(2.5 - 2)^2 + (2.5 - 3)^2 + (4 - 2.5)^2}$$

$$distance(Left, Right) = \sqrt{2.75}$$

$$distance(Left, Right) = 1.66$$

Not a match!

Deep Metric Learning

Using deep learning to have a computer come up with a way to measure something.

Training with Triplets

A B C

Face distance (A, B) should be small.

Face distance (A, C) should be large.

OPTOMATICA
THE POWER TO PERFECT

Training with Triplets

A

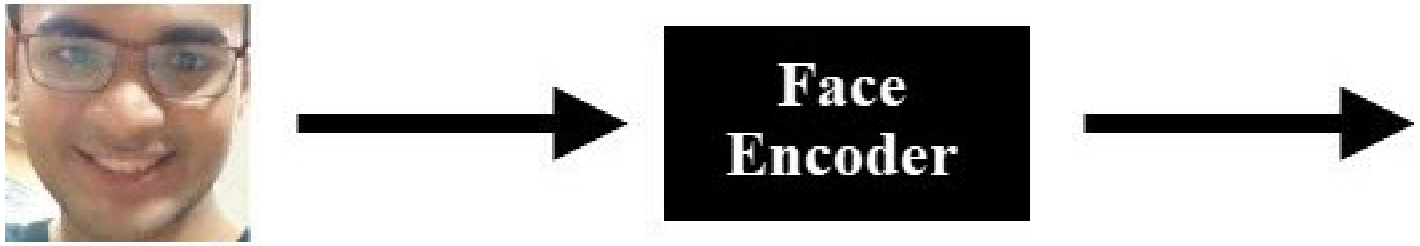
B

C

Face distance (A, B) should be small.

Face distance (A, C) should be large.

Encoding Faces with 128 Points



-0.6948713
0.11813076
0.22286303
0.00477135
-0.0521124
-0.02841697
0.00964657
0.03243792
0.03346656
0.00654369
0.24236825
-0.14097737
-0.31652586
-0.09467196
… etc ….

Calculating Euclidean Distance with 128 Points

$$distance(\mathbf{a}, \mathbf{b}) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_{128} - b_{128})^2}$$

Limitations

- Data and Privacy
- Performance
- Security

Demo ☺