

Face recognition technology



face identification
(surveillance)
arbitrary conditions



face identification
(login)
controlled conditions

SCiFI – A system for Secure Computation of Face Identification

Margarita Osadchy, Benny Pinkas, Ayman Jarrous, Boaz Moskovitch
University of Haifa

Face recognition in surveillance



- **Privacy problem:** the ubiquity of surveillance is a major concern for the public
 - Can be misused to track people regardless of suspicion
 - Can be combined with a universal database linking faces to identities (e.g., drivers' license photos)

We focus on the surveillance problem

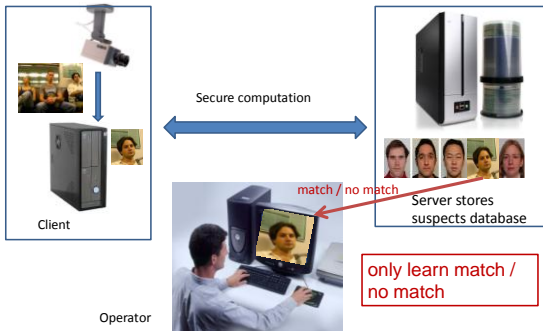


surveillance

Example scenario:

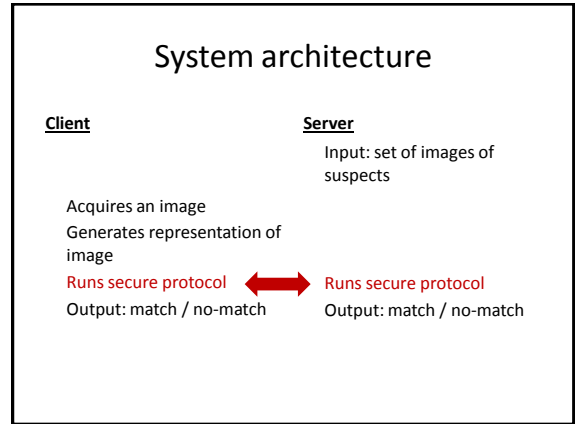
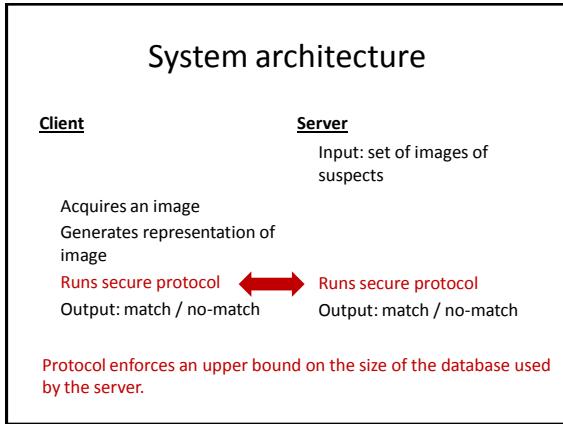
- a government has a list of suspects
- wants to identify them in a crowd

Our approach: protecting the privacy of the public and the confidentiality of the data



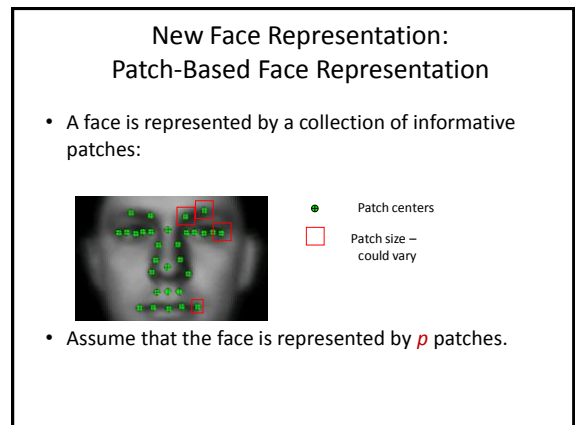
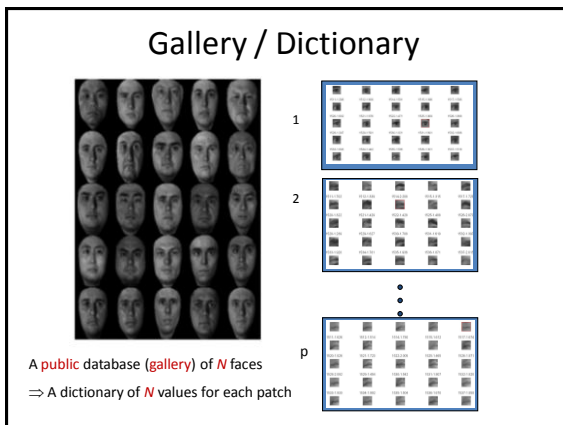
A solution to the privacy concern

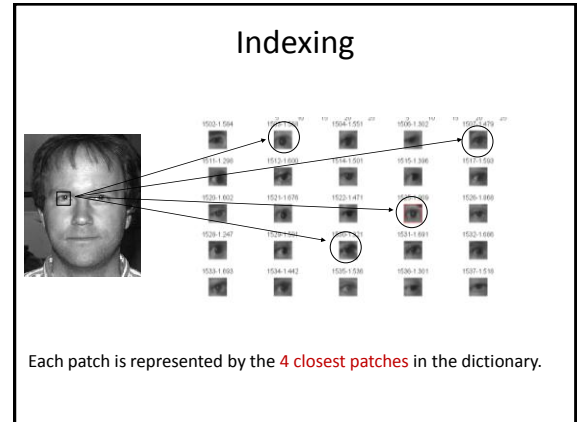
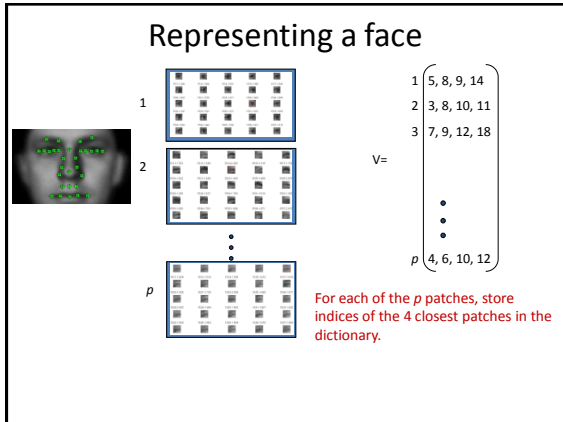




- ### Our Contributions
- A new and unique **face identification algorithm**
 - Specifically designed for secure computation
 - Has state-of-the-art **recognition** performance
 - Assumes only a **single** image is known per suspect
 - A **secure protocol** for computing face identification
 - **SCIFI** - A system implementing the protocol
 - Previous work [EFGKLT09]: secure computation of the well known Eigenfaces face recognition algorithm.
 - Performance of eigenfaces is inferior to state-of-the-art.
 - The secure protocol is less efficient than ours.

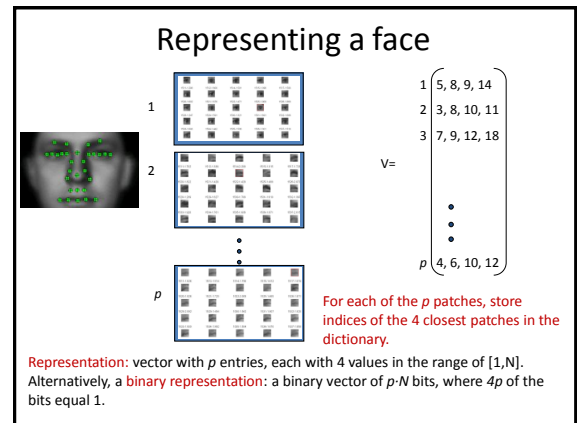
- ### The Problem
- Exact / fuzzy match
 - Secure computation of **exact** matches is well known.
 - Face identification is **fuzzy**. A match is between *close*, but *not identical*, images.
 - Continuous / discrete math
 - Face recognition algorithms use **continuous** face representations, and complex measures of similarity.
 - Secure computation is always applied to **discrete** numbers. Best with linear operations.
 - Simple quantization of face recognition algorithms results in poor performance.





Similarity between faces

- We define the difference between faces as the **set difference** between their representations
 $\Delta(A,B) = |A \cup B| - |A \cap B|$
- Set difference \equiv **Hamming distance between binary representation of faces**
- Secure computation of Hamming distance is easy [JP09]



The protocol in a nutshell

(details and proof in the paper)

- Inputs are vectors $w = w_0, \dots, w_{m-1}$; $w' = w'_0, \dots, w'_{m-1}$.
- Client sends $E(w_0), \dots, E(w_{m-1})$
- Server uses homomorphic properties
 - To compute $E(w_0 \oplus w'_0), \dots, E(w_{m-1} \oplus w'_{m-1})$
 - To sum these values and obtain $E(d_H(w, w')) = E(d)$
- Server chooses random R ; sends $E(d+R)$ to client
- Client decrypts $E(d+R)$, reduces the result mod $m+1$.
- Both parties run a **1-out-of-(m+1) OT**, where client learns 1 if Hamming distance $<$ threshold.

Cryptographic Protocol

- Functionality:**
 - Client and server each have a binary vector representing a face.
 - Output 1 iff Hamming distance $<$ threshold.
- Tools**
 - Additively homomorphic encryption
 - Given $E(x)$, $E(y)$ can compute $E(x+y)$
 - Oblivious transfer
 - A two-party protocol where receiver can privately obtain one of two inputs of a sender

Online overhead

- A face is represented by a 900 bit vector.
- **Overhead after the client captures an image:**
 - Client sends 900 bits to server
 - For every image in server's database
 - Server performs 450 homomorphic additions
 - Server sends a single encryption to client
 - Client decrypts the encrypted value
 - Run a *preprocessed* OT: client sends 8 bits to server; server sends 180 bits to client.

Optimizations

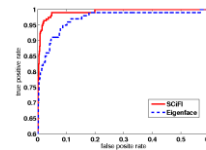
- **Main goal:** minimize *online* latency, to identify suspects in real time.
- **Methods used:**
 - Change protocol s.t. oblivious transfer and most communication can be done **before** image is recorded.
 - Prefer more efficient homomorphic operations
addition \ll *encryption* $<$ *subtraction*

Implementation

- **Face recognition** part (generating representations of images)
 - Implemented in **Matlab**, ran using Matlab Java builder.
- **Cryptographic** protocol
 - Implemented in **Java**, using Paillier and ElGamal based OT.
- **Timing on Linux servers:**
 - ~0.3 sec to compare to a single image in the database
 - An Implementation in C will be much faster
 - Easily parallelizable

Recognition experiments

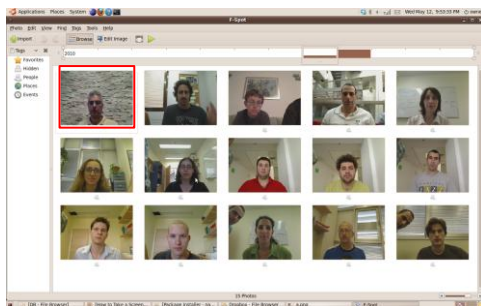
- Ran experiments with *standard databases* used by the face recognition community.
- Tested **robustness** to illumination changes, small changes in pose, and partial occlusions.



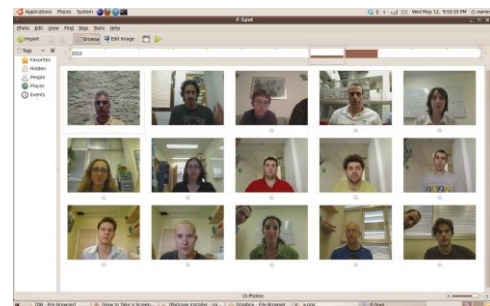
Robustness compared to Eigenfaces

Robustness to partial occlusions

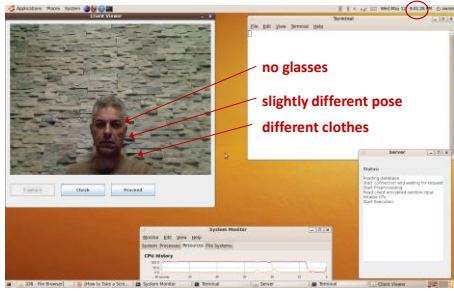
The suspect



The database



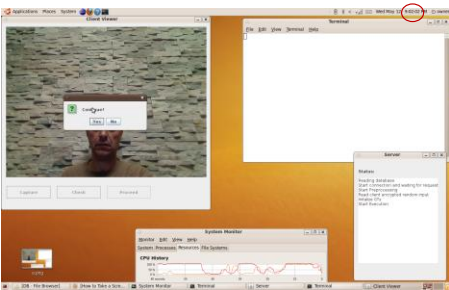
An image is obtained by the client



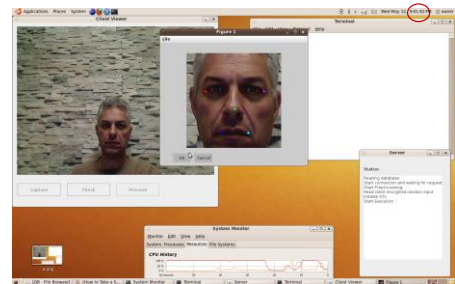
The suspect



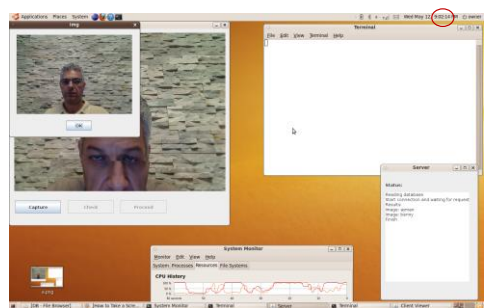
Face representation is ready



Facial features are recognized



Secure protocol is run, a match is found



Live demo available upon request

Conclusions

- **Goal:** Face recognition based surveillance, respecting subjects privacy.
- **Means:**
 - A new and unique face identification algorithm
 - State of the art robustness
 - Suitable for secure computation
 - A secure protocol with optimized online runtime
 - Experiments verifying robustness and performance