Iris Recognition

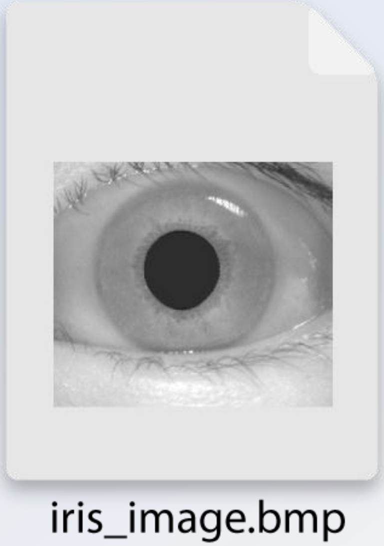
Why Iris Recognition?

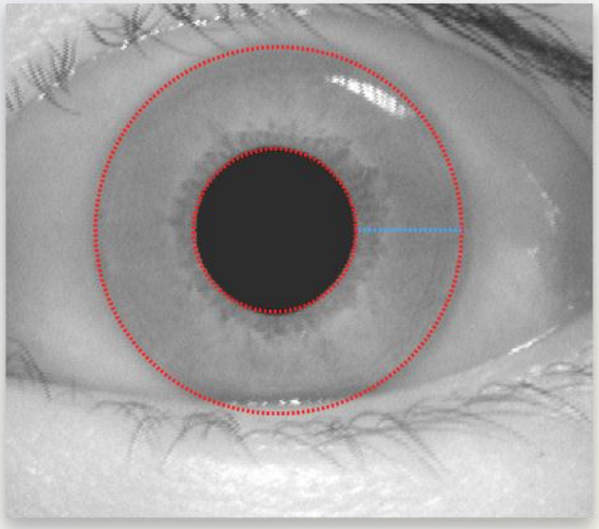
Biometric identification

Iris recognition

- ✓ Stable - unchanging over lifetime
- ✓ Unique - huge pattern variability
- ✓ Reliable

Primary Tasks



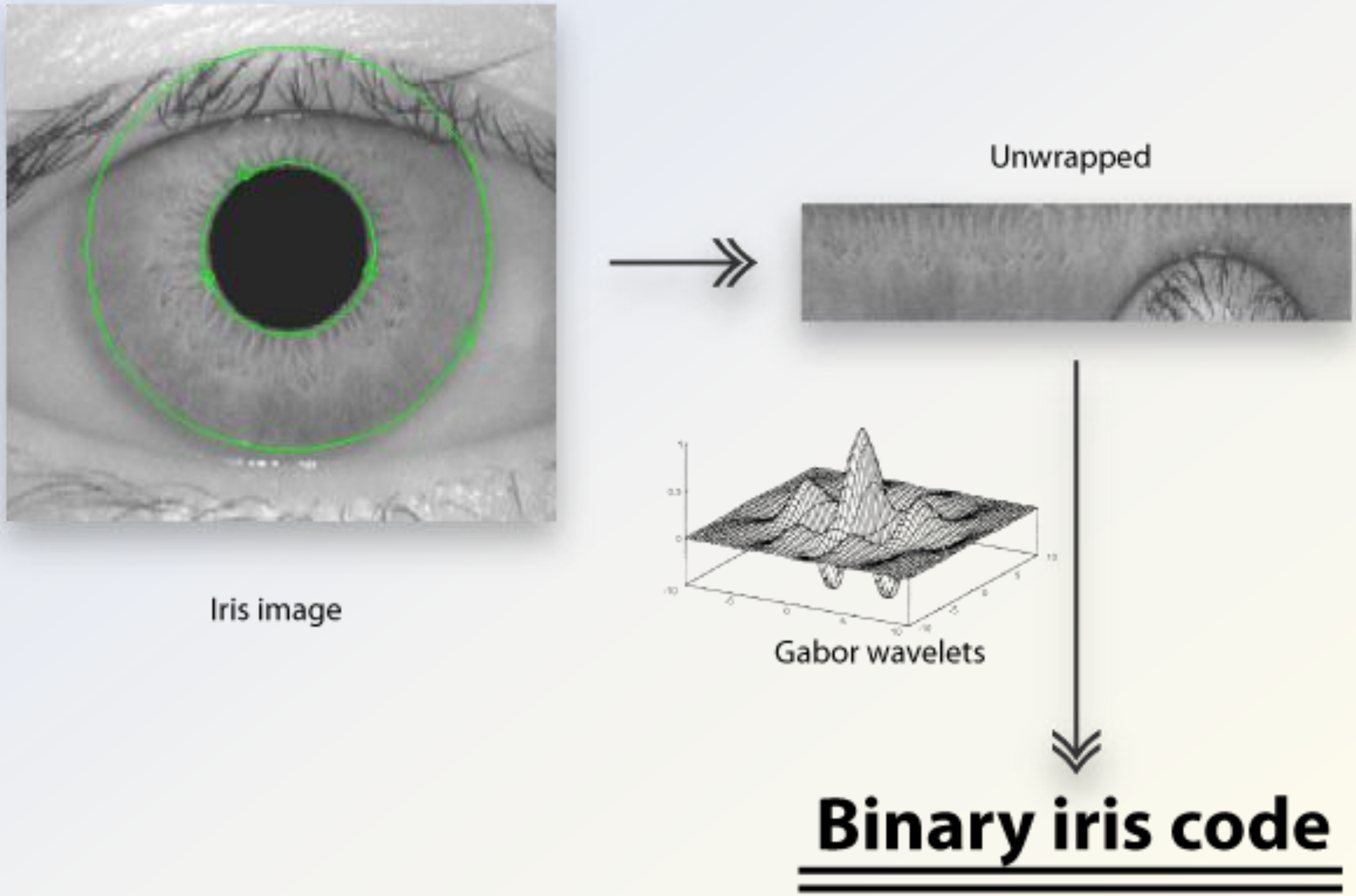


Display image of an eye and identify the pupil and Iris edge

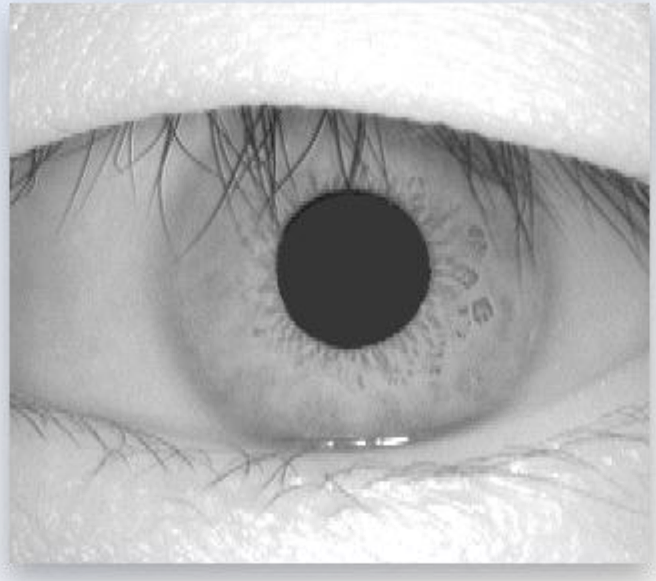
Two possible methods:

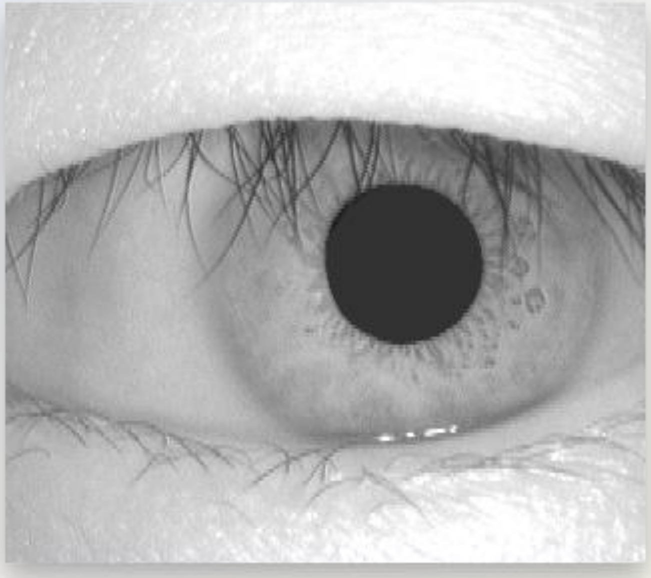
a. User input
b. Auto-detection

Encoding the Iris Pattern



Comparing the Iris Codes



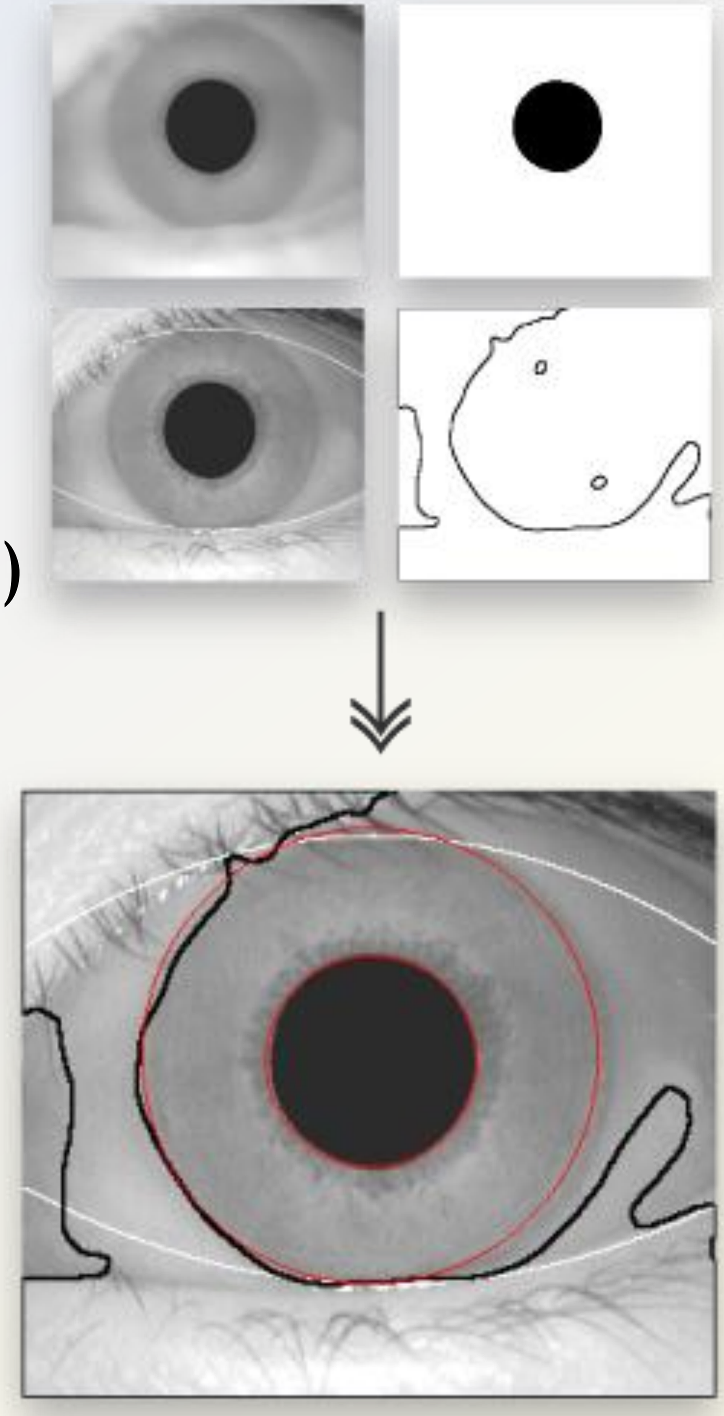


0110011101110011011001110
1100011011000010111001101
1001001101110011011001110
1100011011000010111001101
1001001101110011011001110
1100011011000010111001101

?
≈

0110011101111100011001110
1100011011000010111001101
1001001111110011011001110
1100011011000010111001101
1001001101110011011001110
1100011011000000111001101

Iris Auto-Detection



Process for identifying circles.

Iris and pupil concentric circles (roughly)

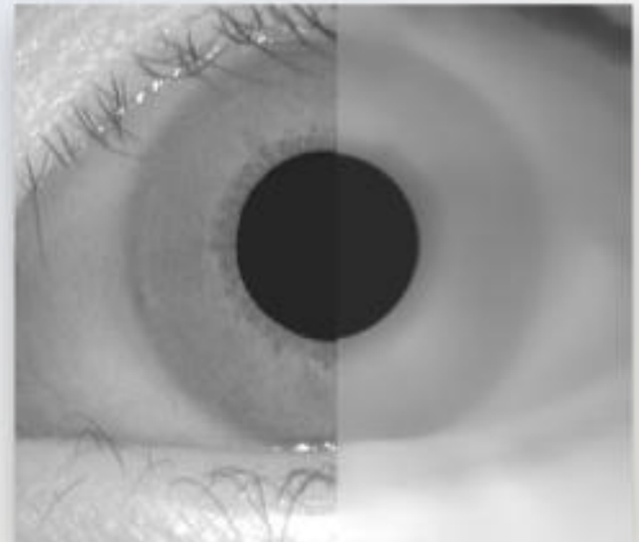
Several steps involved:

- Reduce noise
- Threshold image
- Edge-detect
- Identify circle

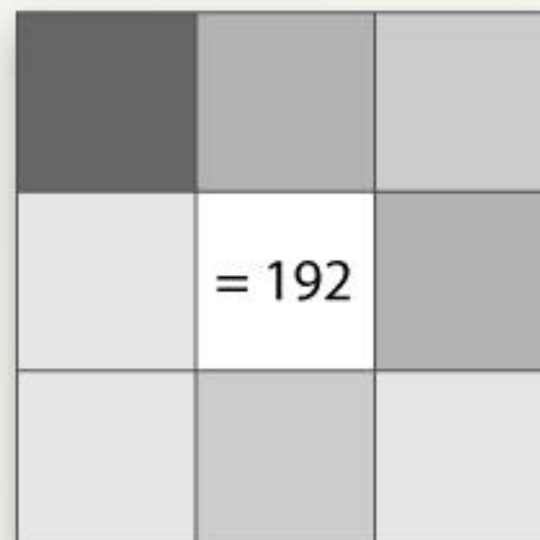
Median Filter

Used for noise reduction

Reduces pixel complexity without losing edge fidelity



{ 94, 129, 155, 192, 192, 192, 202, 204 }



Sobel Filter

Kernel-based edge detection

Uses first derivative of image

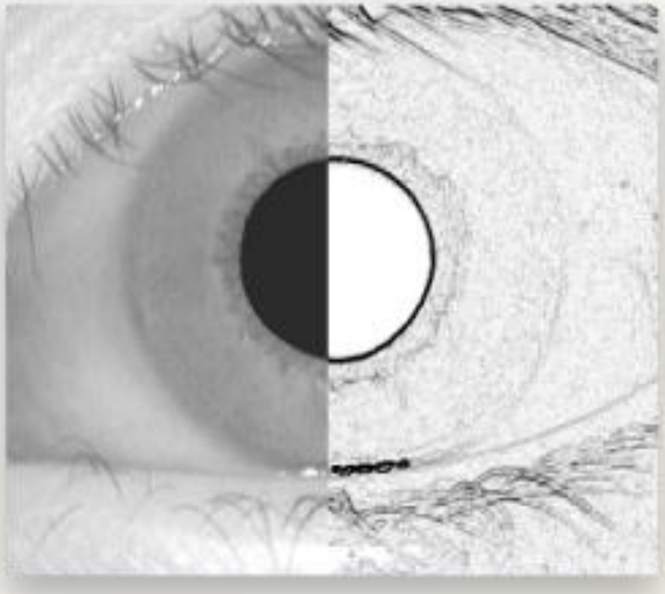
-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

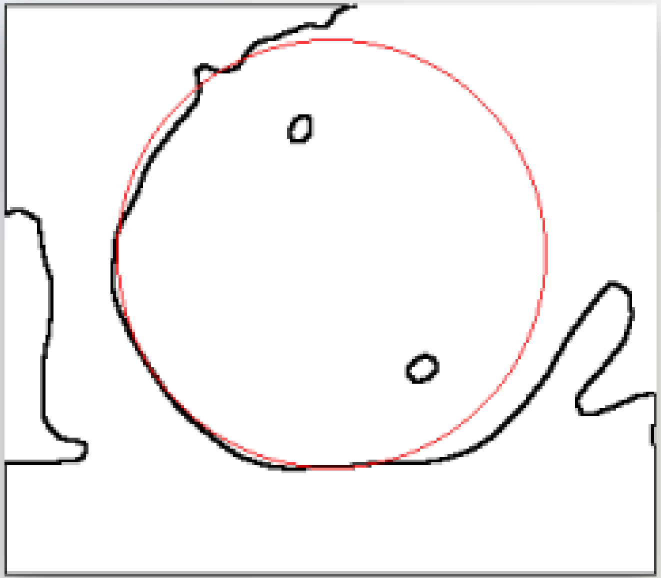
$$|G| = \sqrt{Gx^2 + Gy^2}$$



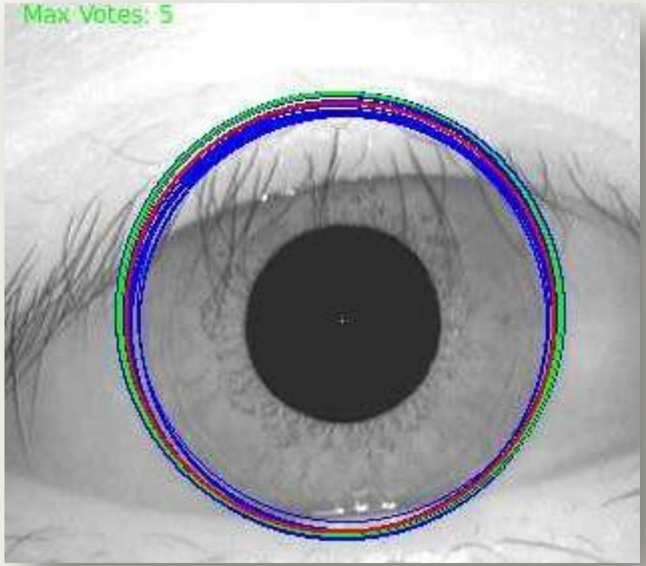
too much edge data

Hough Transform for Locating Circles

- ✓ Resilient to noise
- ✓ Adaptable to variable input
- ✓ Provides a good best-fit



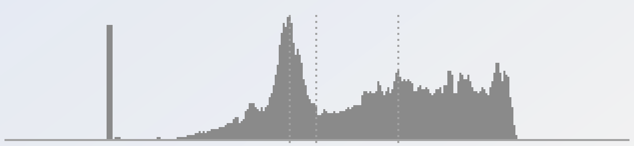
- × Computationally quite expensive
- × Requires experimentation for best results



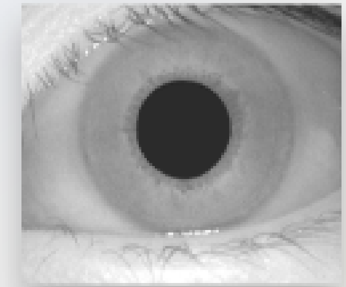
Identifying the Pupil

Most reliable, uniform, circular artefact in image
Used to guide search for iris in next step

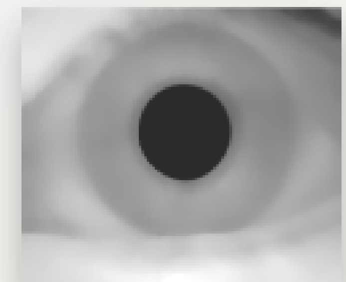
1. Median: reduce noise
2. Threshold: remove irrelevant information based on histogram analysis.



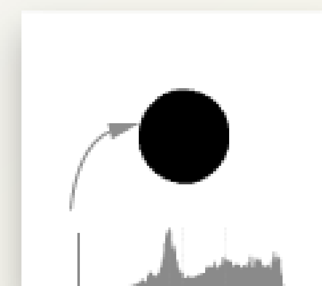
3. Sobel: detect edges
4. Hough Transform: estimate circle identifying pupil



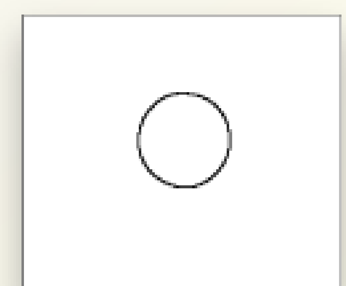
median filter



threshold



edge detect

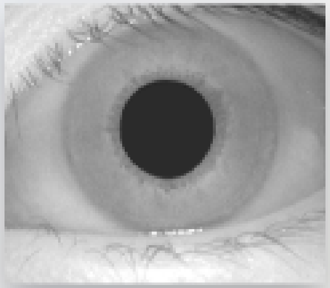


Identifying the Iris

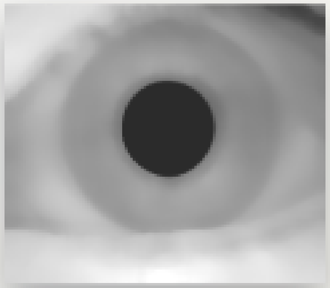
Similar method to the pupil detection

Hough parameter space is constrained using pupil location and dimensions

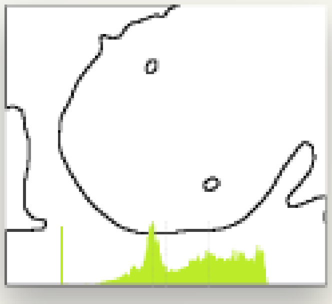
Result is a good estimation of iris bounds in a matter of milliseconds



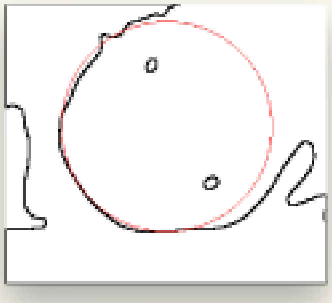
median filter



thresh + edge



Hough



Eyelids

Need to eliminate useless bits from the iris

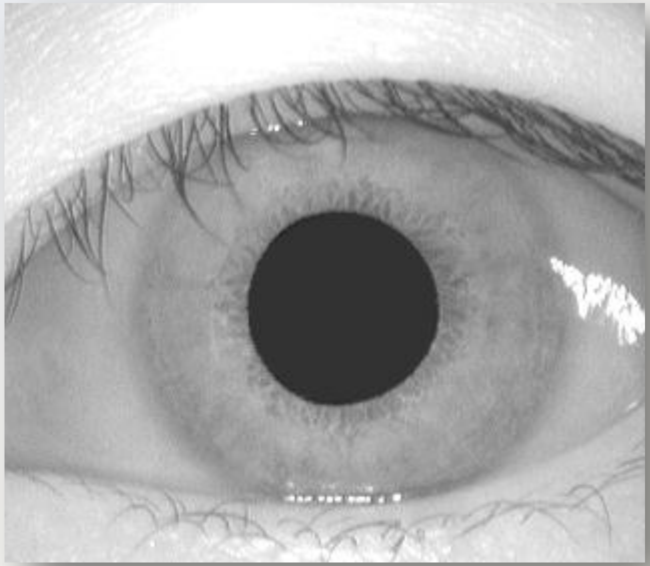
Generate a mask to mark areas as bad bits

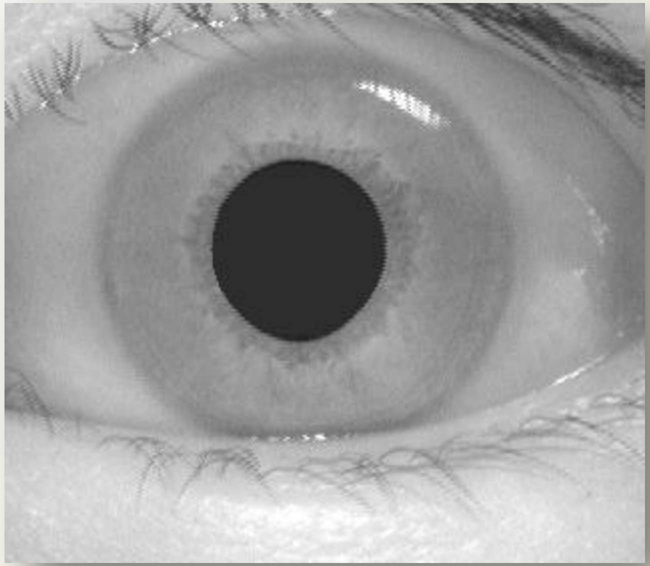
Eyelids generally have higher intensity than the iris and sclera

Specular Highlight Removal

Highlights can encroach on iris data and affect phase information

High intensity pixels in the image can be masked and safely discarded in the encoding phase



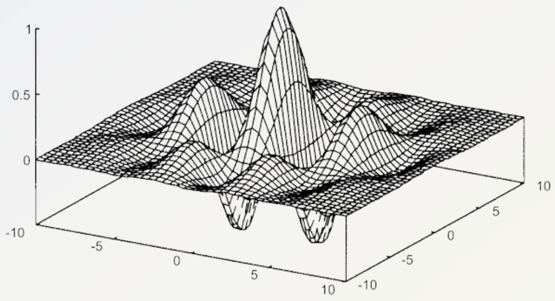


Encoding the Iris: Gabor Filters

Gabor wavelets are used for encoding the iris

Phase quantisation of the iris pattern

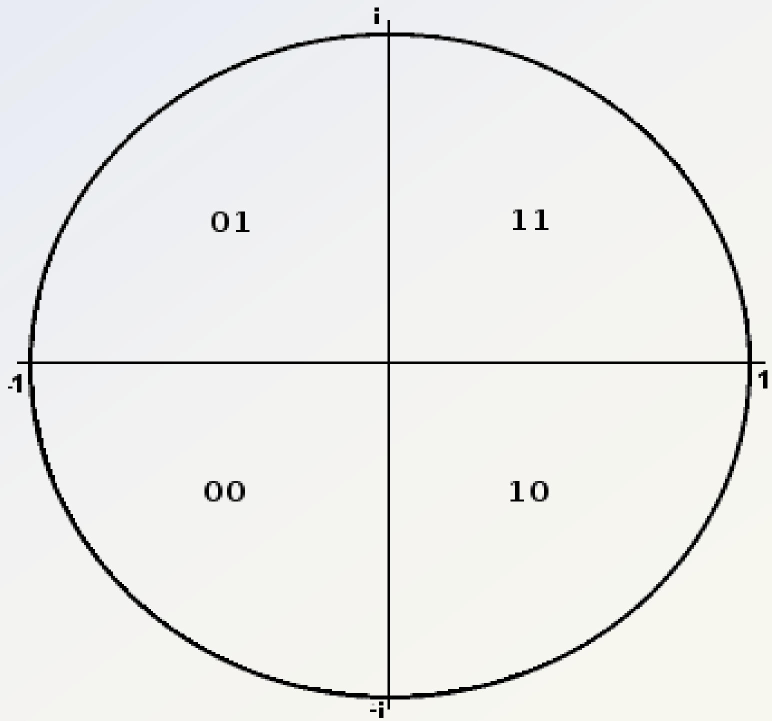
Encode based on the signs of real and imaginary parts



Gabor Wavelet plotted on complex plane

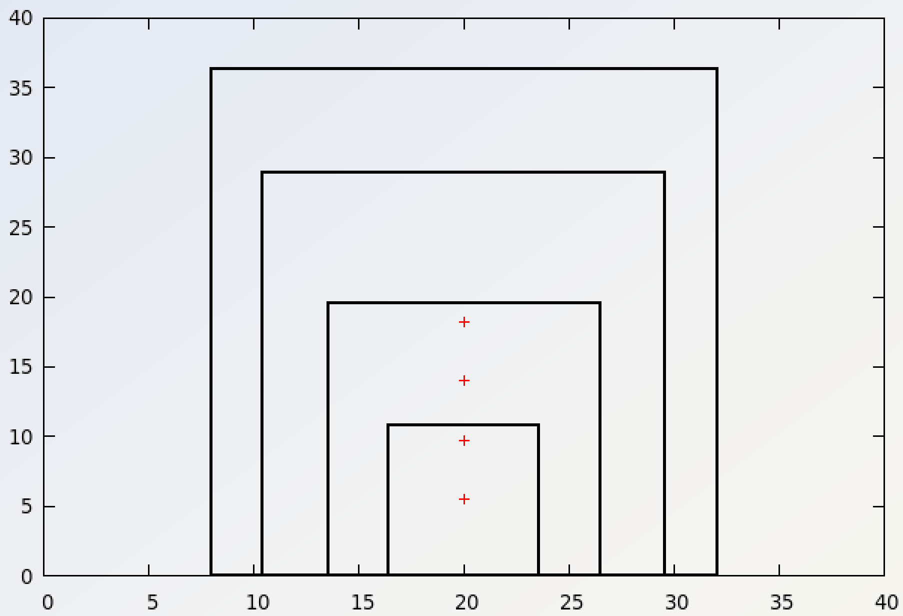
Gabor Wavelets

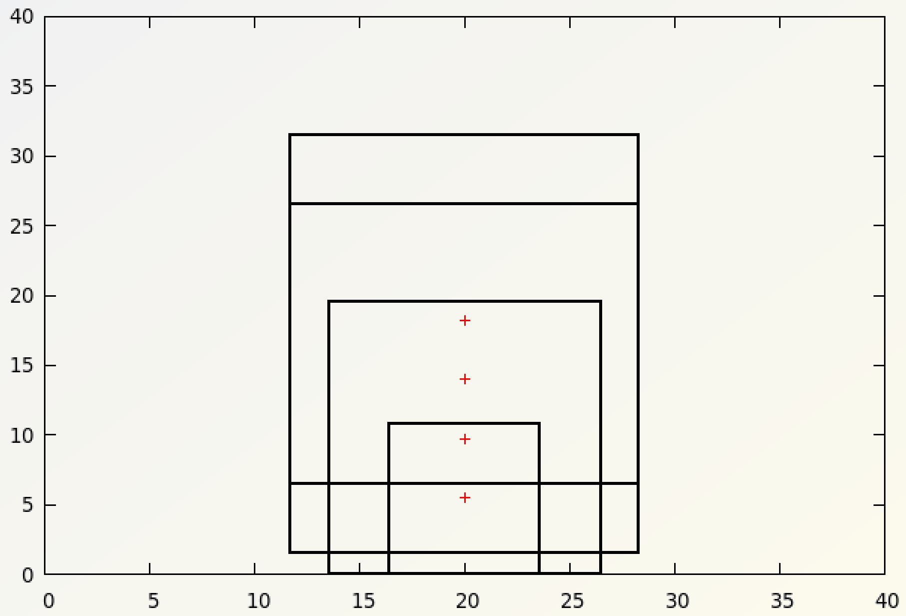
Returns 2 bits depending on the phase quadrant



Gabor Filter Placement and Size

- 2048 bit string
- Filters applied to 256 angular directions





Comparing Bitcodes

Calculate Hamming Distance between bitcodes

- Method for comparing strings of equal length

Iris Bitcode 1	Iris Bitcode 2
10110110001001011011	11010110110001111011

20 bits, 6 differences → Hamming distance of 0.3 (6 / 20)

Comparing Bitcodes with Mask

$$H = \frac{\|(codeA \otimes codeB) \cap maskA \cap maskB\|}{\|maskA \cap maskB\|}.$$

<table>
<tr><th>Iris Bitcode</th><th>Mask</th><th>Mask A or Mask B</th><th>Useful Comparable Bits</th></tr>
<tr><td>1010110011</td><td>1100111111</td><td rowspan="2">1100001111</td><td>10 _ _ _ _ 0011</td></tr>
<tr><td>1000010011</td><td>1111001111</td><td>10 _ _ _ _ 0011</td></tr>
</table>

Codes match on useful (iris information) bits

Design Decisions

C++

- Extremely fast
- Flexible

Qt

- GUI toolkit
- Cross-platform
- Many useful classes

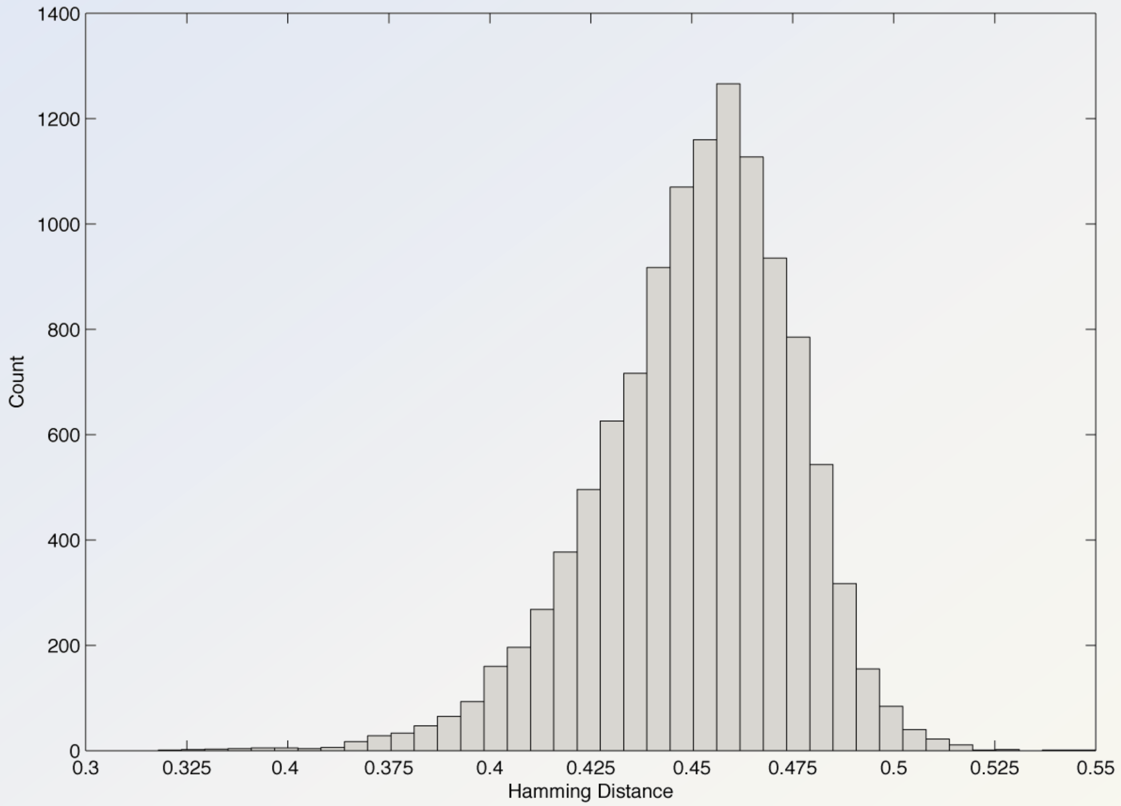
http://qt.nokia.com/

Ir Demonstration!

Results

- 0 false positives in ~ 35,000 comparisons
- ~70% success on iris auto-detection, resulting in an overall 70% match rate
- Inaccuracy due to auto-detection failure
- Manual point selection improves the rate

Hamming Distances



Resembles a binomial distribution with $\mu = 0.45$, $\sigma = 0.024$

Future Work

- Improve auto-detection of iris with Otsu method
- Eyelash detection and removal
- Improve eyelid detection using Hough Transform
- Progress application to a client/server implementation

Source Code

Code freely available under the GPL licence:

http://projectiris.co.uk