Set 3: Image Transforms

Lecturer Arto Kaarna

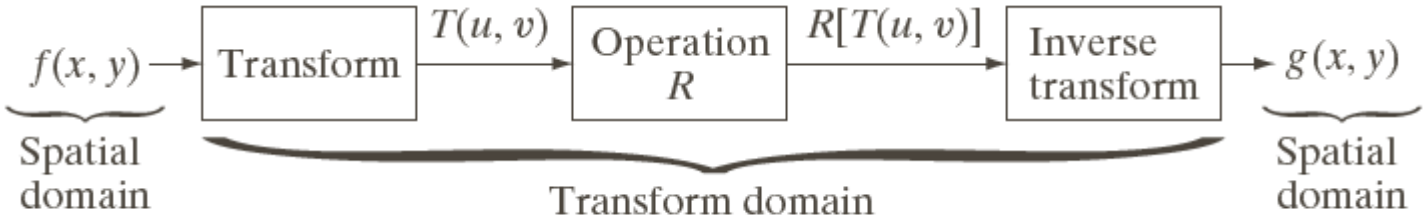
Lappeenranta University of Technology (LUT)
School of Engineering Science (LENS)
Machine Vision and Pattern Recognition (MVPR)

Arto.Kaarna@lut.fi
http://www.lut.fi/web/en/school-of-engineering-science/research/machine-vision-and-pattern-recognition

Contents

- Fourier Transform
  - Distance transform (as a simple introduction to the topic)
  - Discrete Fourier Transform
  - Properties of 2-D Fourier Transform
    - Separability, periodicity, translation, rotation, scaling, convolution, correlation
  - Fast Fourier Transform
  - Fourier Transform for image processing and analysis
- Gabor filtering
  - 2-D Gabor filter
  - Applications
- Other transforms

Transforms in General



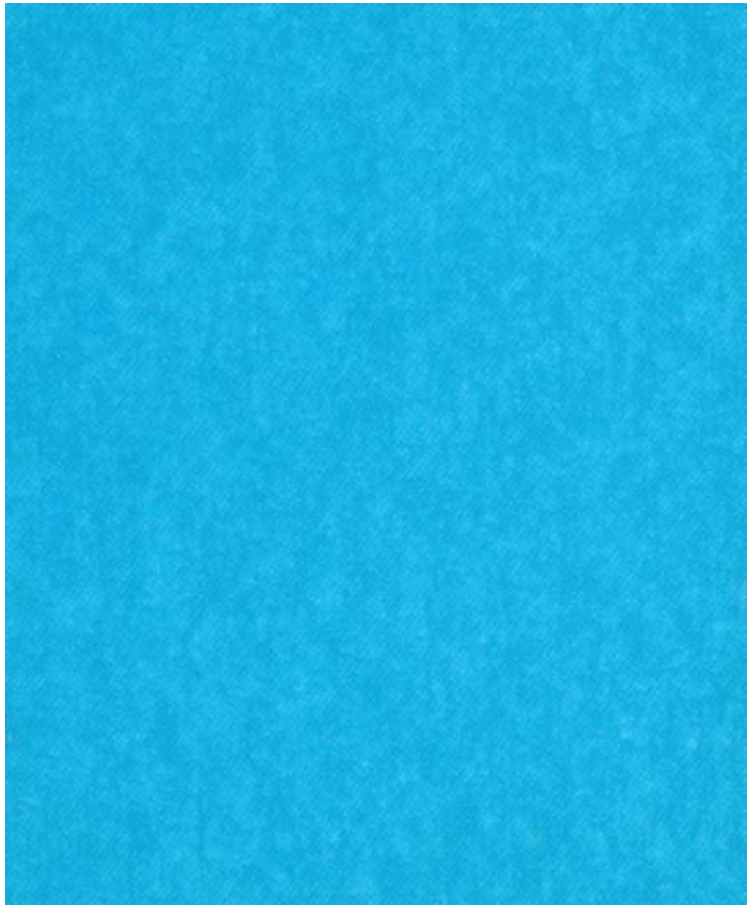
- With images there is the transform from *spatial domain* to *transform domain.*

$$T(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\, r(x,y,u,v)$$

  - E.g. time domain/pixel domain to frequency domain
- Most useful transforms have also an *inverse transform*.

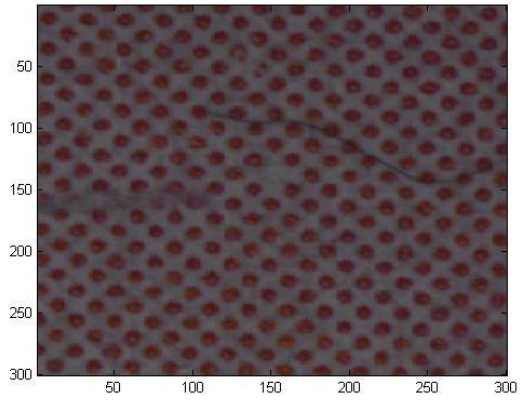
FT in Image Analysis: Mottling

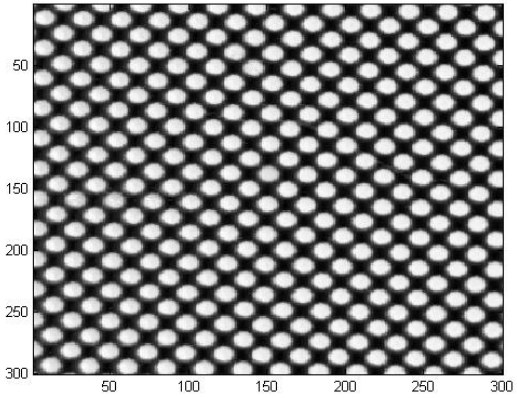
- Extracting frequency information from the image
  - Size of patterns: small-scale and large-scale variations in an image, e.g. the evenness of monochrome print = mottling.

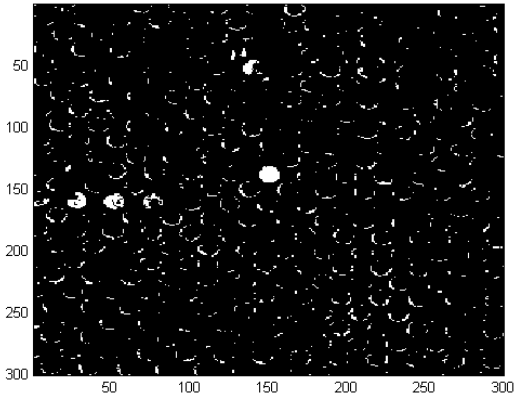


FT in Image Analysis: Heliotest

- Texture and other repetitive patterns
- FT is used to find the ”regular” image (containing the ”full” repetitive pattern)
- ”irregularities” can be detected in the difference image

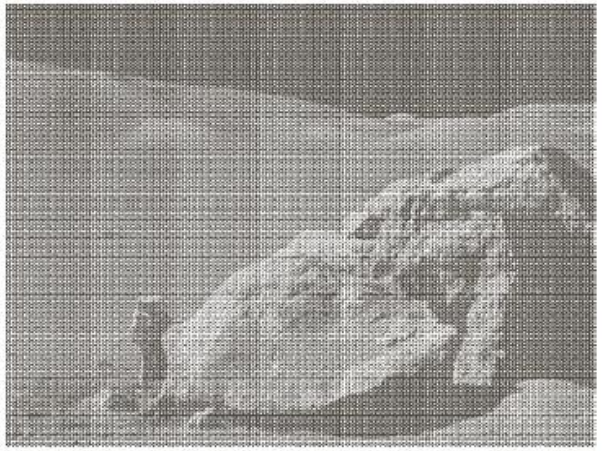


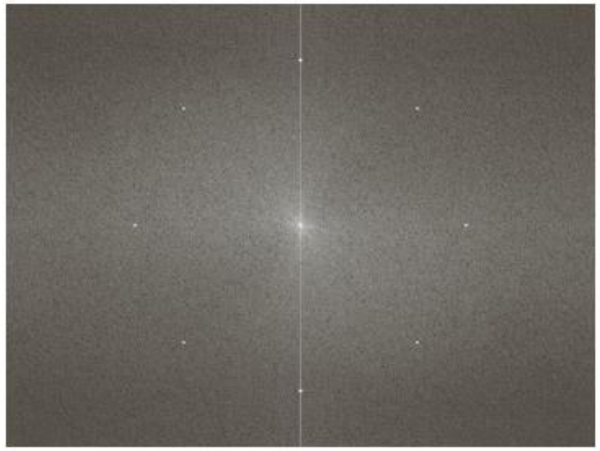


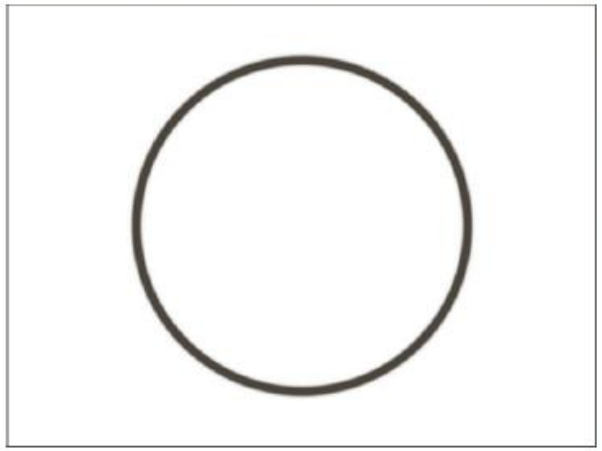


Example on FT

- Operation R: removing noise







a b
c d

FIGURE 2.40
(a) Image corrupted by sinusoidal interference. (b) Magnitude of the Fourier transform showing the bursts of energy responsible for the interference. (c) Mask used to eliminate the energy bursts. (d) Result of computing the inverse of the modified Fourier transform. (Original image courtesy of NASA.)

Image Transforms

- Fourier Transform
- Cosine
- Sine
- Hadamard
- Haar
- Slant
- Karhunen-Loeve
- Fast KL
- Sinusoidal
- SDV
- etc.

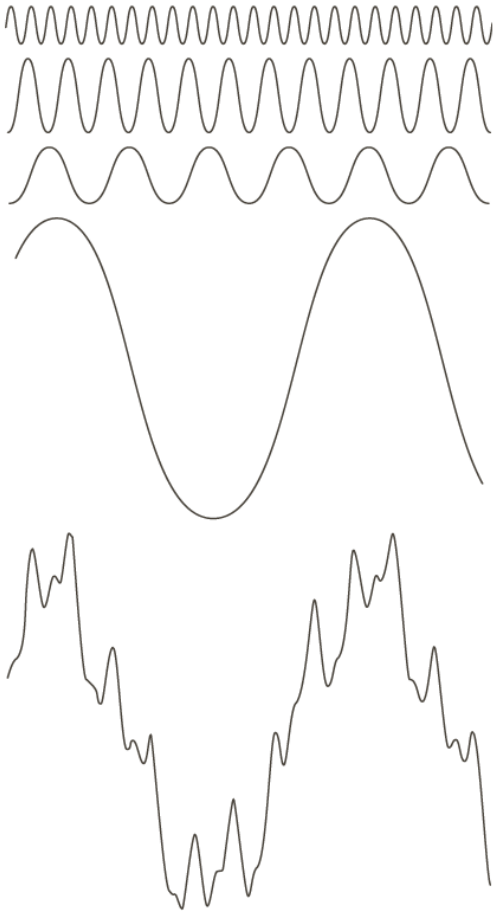
DFT/unitary DFT	Fast transform, most useful in digital signal processing, convolution, digital filtering, analysis of circulant and Toeplitz systems. Requires complex arithmetic. Has very good energy compaction for images.
Cosine	Fast transform, requires real operations, near optimal substitute for the KL transform of highly correlated images. Useful in designing transform coders and Wiener filters for images. Has excellent energy compaction for images.
Sine	About twice as fast as the fast cosine transform, symmetric, requires real operations; yields fast KL transform algorithm which yields recursive block processing algorithms, for coding, filtering, and so on; useful in estimating performance bounds of many image processing problems. Energy compaction for images is very good.
Hadamard	Faster than sinusoidal transforms, since no multiplications are required; useful in digital hardware implementations of image processing algorithms. Easy to simulate but difficult to analyze. Applications in image data compression, filtering, and design of codes. Has good energy compaction for images.
Haar	Very fast transform. Useful in feature extracton, image coding, and image analysis problems. Energy compaction is fair.
Slant	Fast transform. Has "image-like basis"; useful in image coding. Has very good energy compaction for images.
Karhunen-Loeve	Is optimal in many ways; has no fast algorithm; useful in performance evaluation and for finding performance bounds. Useful for small size vectors e.g., color multispectral or other feature vectors. Has the best energy compaction in the mean square sense over an ensemble.
Fast KL	Useful for designing fast, recursive-block processing techniques, including adaptive techniques. Its performance is better than independent block-by-block processing techniques.
Sinusoidal transforms	Many members have fast implementation, useful in finding practical substitutes for the KL transform, analysis of Toeplitz systems, mathematical modeling of signals. Energy compaction for the optimum-fast transform is excellent.
SVD transform	Best energy-packing efficiency for any given image. Varies drastically from image to image; has no fast algorithm or a reasonable fast transform substitute; useful in design of separable FIR filters, finding least squares and minimum norm solutions of linear equations, finding rank of large matrices, and so on. Potential image processing applications are in image restoration, power spectrum estimation and data compression.

Fourier Transform

- Jean Baptiste Joseph Fourier 1807:

  "Any periodic function can be expressed as the sum of sines and/or cosines of different frequencies, each multiplied by a different coefficient."

- The sum is nowadays called a Fourier series
- An image is a 2D signal



Fourier Transform

- Let *f(x)* be a continuous function of a real variable *x*. Then the Fourier transform of *f(x)* denoted *F{f(x)}* is defined by

$$\mathcal{F}\{f(x)\} = F(u) = \int_{-\infty}^{\infty} f(x) e^{-\mathrm{j}2\pi ux} dx \quad \text{where } j = \sqrt{-1}$$

- Given *F(u)*, *f(x)* can be obtained by the inverse Fourier Transform

$$\mathcal{F}^{-1}\{F(u)\} = f(x) = \int_{-\infty}^{\infty} F(u) e^{\mathrm{j}2\pi ux} du$$

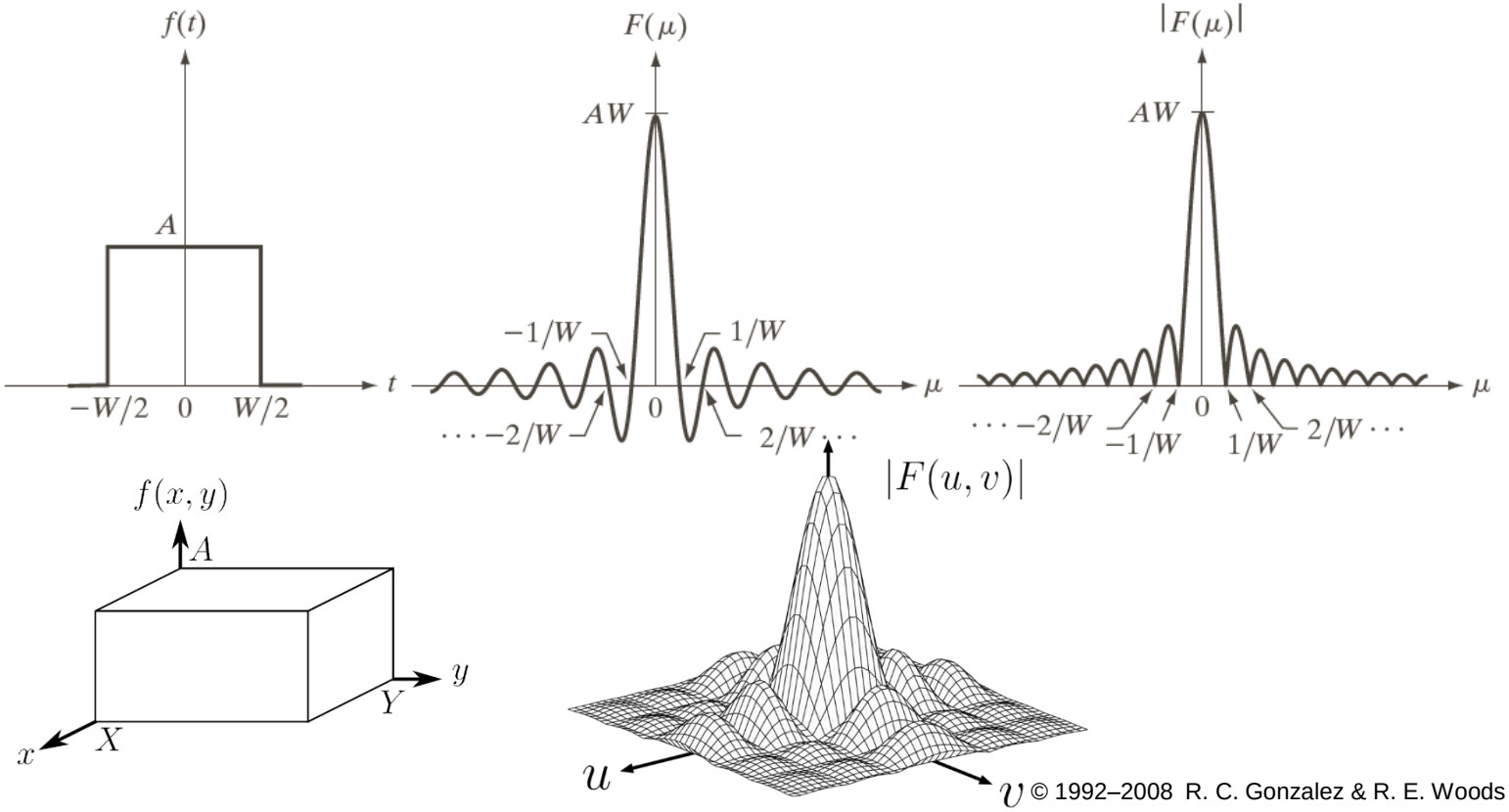
- The FT of a real function is often complex (sum of real and complex part), and its magnitude is the Fourier spectrum:

$$|F(u)| = \sqrt{\Re^2(u) + \Im^2(u)}$$

- See e.g. http://miac.unibas.ch/BIA/

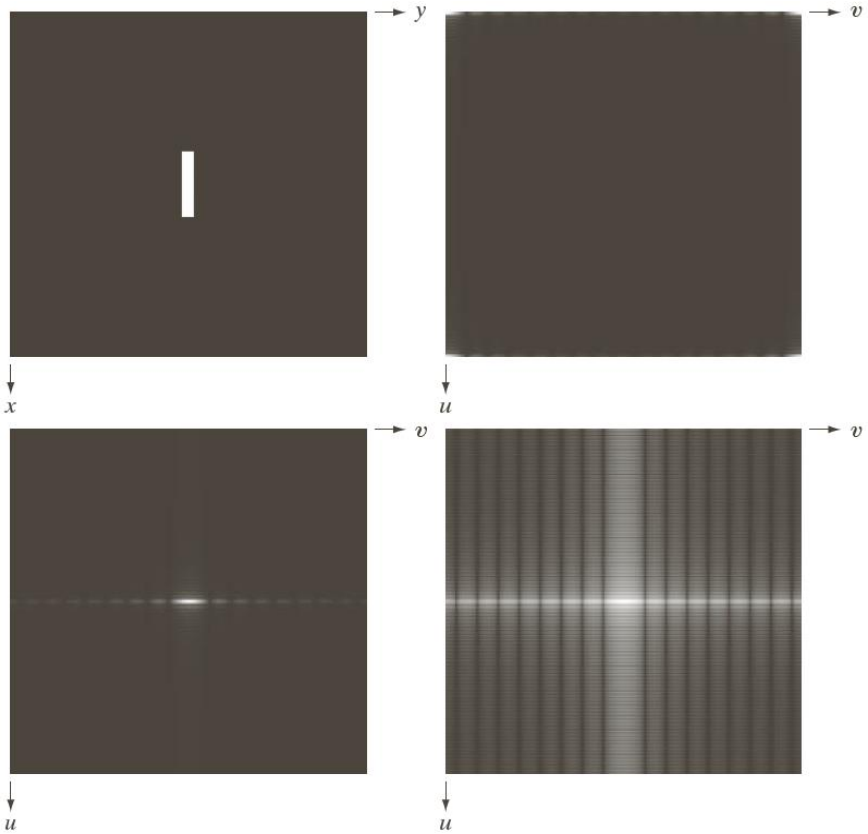
Fourier Transform

- Signal and the corresponding spectrum



Fourier Transform

- Typical presentation of the spectrum of a rectangle

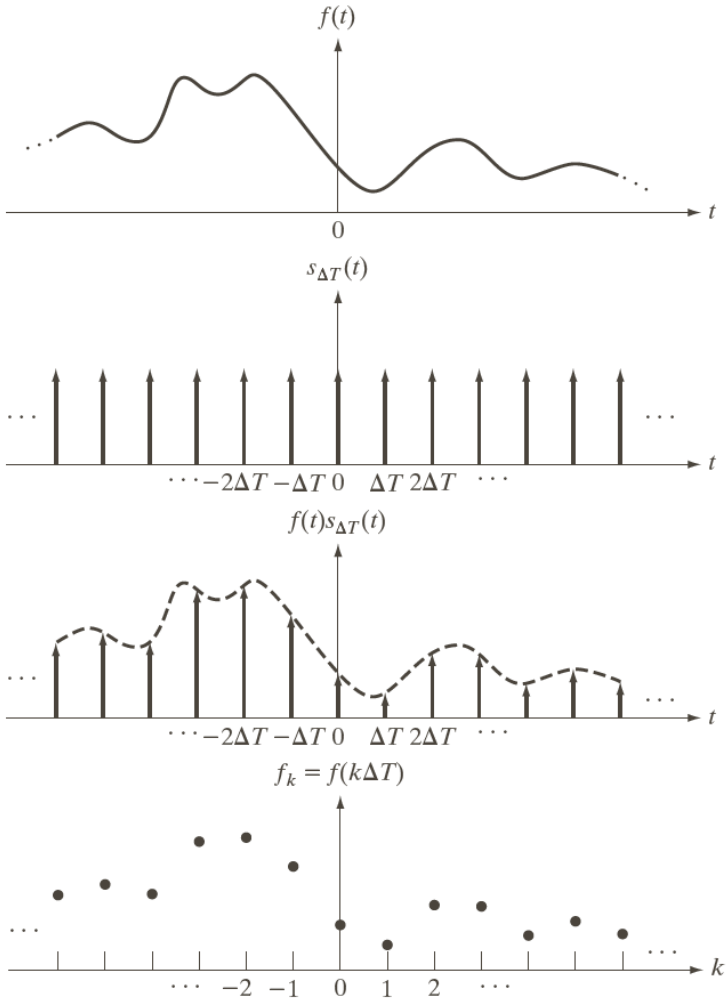


Discrete Fourier Transform (DFT)

- In the previous the variables were continuous (*x,y,u,v)*
- For images, discrete variables are more realistic
  - Sampling in time domain
- Continuous transform of a sampled signal

$$\tilde{F}(\mu) = F(\mu) * S(\mu)$$

$*$ stands for convolution



Discrete Fourier Transform (DFT)

- FT of a sampled one-dimensional signal is

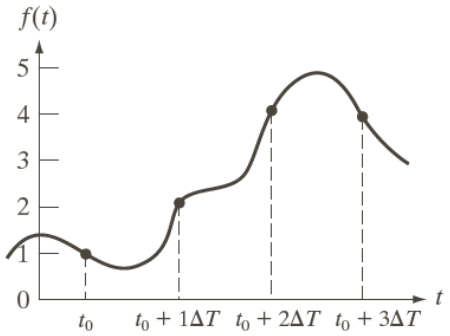
$$\tilde{F}(\mu) = \int_{-\infty}^{\infty} \tilde{f}(t) e^{-j2\pi\mu t}\, dt$$

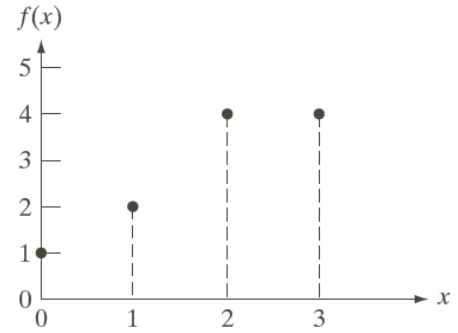
- And

$$\tilde{F}(\mu) = \sum_{n=-\infty}^{\infty} f_n e^{-j2\pi\mu n\Delta T}$$

- Now the time domain is discretized as f_n
- To obtain DFT the frequencies are also sampled

$$\mu = \frac{m}{M\Delta T}$$





2D DFT

- Now this leads to DFT

$$F_m = \sum_{n=0}^{M-1} f_n e^{-j2\pi mn/M}$$

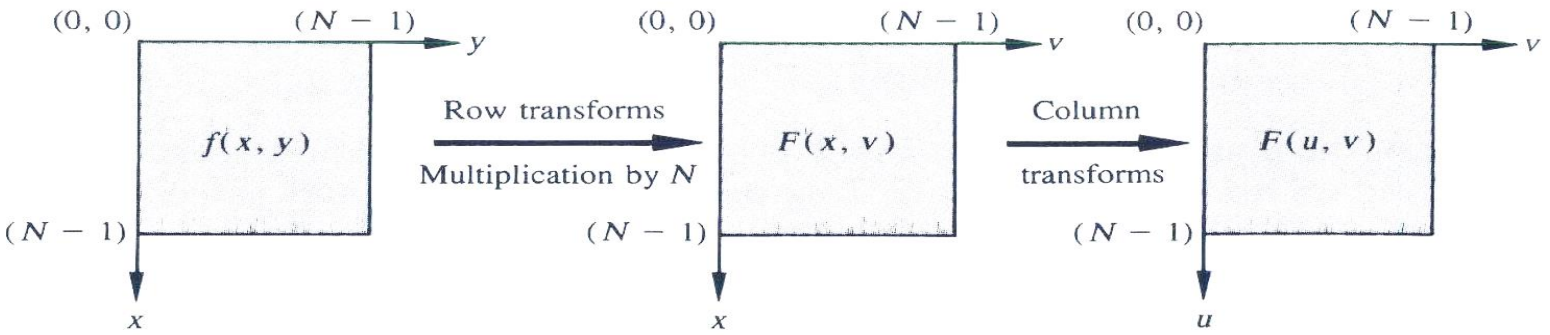
- In image processing the coordinates are (x, y) and (u, v)
- Now DFT becomes

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

- where $u = 0,1,2,\ldots,M-1$ and $v = 0,1,2,\ldots,N-1, M$ and N are the dimensions of the image.

Properties of 2D DFT

- Separability
  - 2-D can be computed as series of 1-D transforms



Properties of 2D DFT

- Periodicity: $F(u,v) = F(u + k_1 M, v) = F(u, v + k_2 N)$

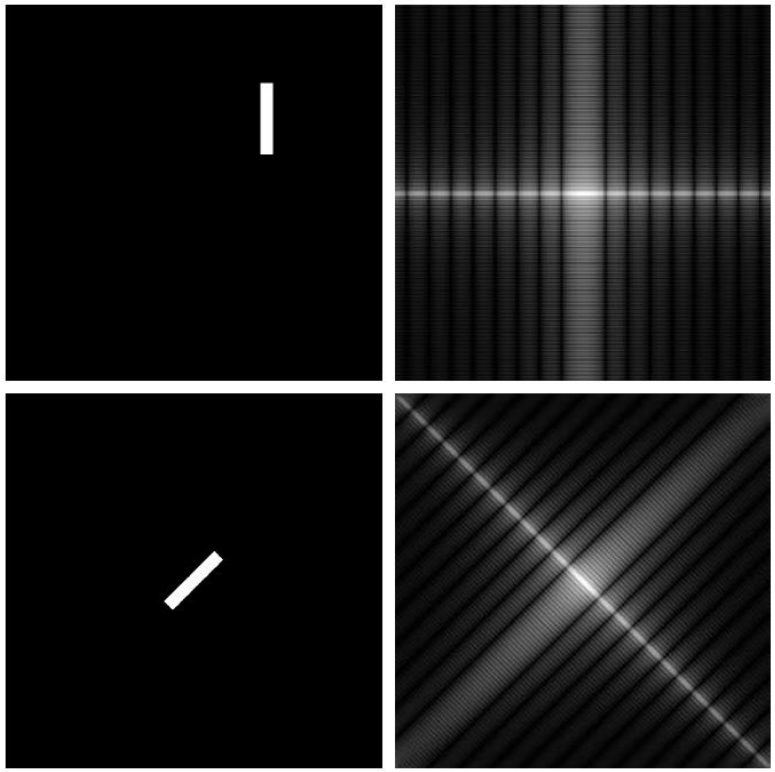
- Translation: $f(x,y)e^{-j2\pi(\frac{u_0 x}{M}+\frac{v_0 y}{N})} \leftrightarrow F(u - u_0, v - v_0)$
- Rotation: $f(r, \theta + \theta_0) \leftrightarrow F(\omega, \varphi + \theta_0)$

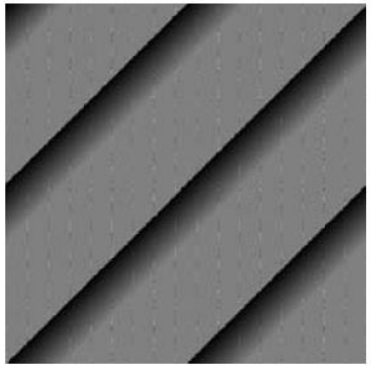
- Convolution: $f(t)h(t) \leftrightarrow H(\mu) * F(\mu)$

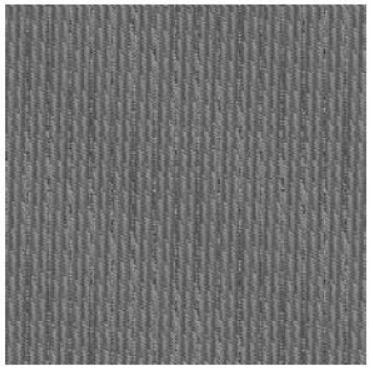
  $f(t) * h(t) \leftrightarrow H(\mu)F(\mu)$

  - Spatial filtering in the frequency domain
- Even and odd parts of a complex variable

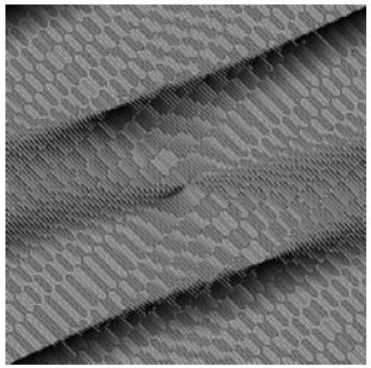
Examples of 2D DFT





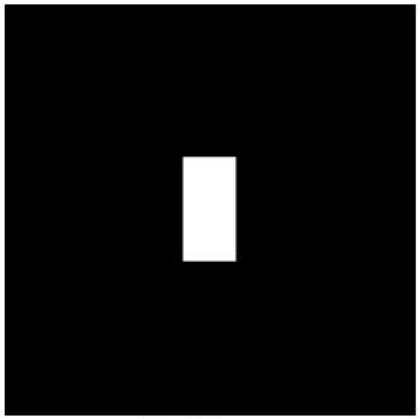


a b c

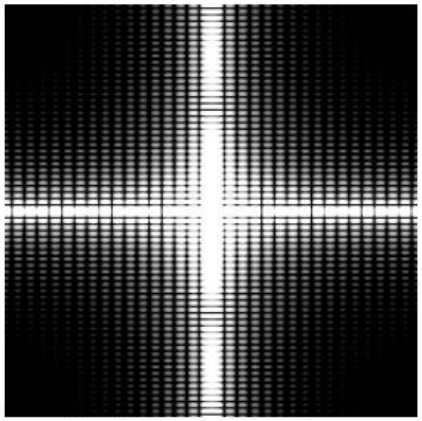


Phase from Original (a) Translated (b) Rotated (c)

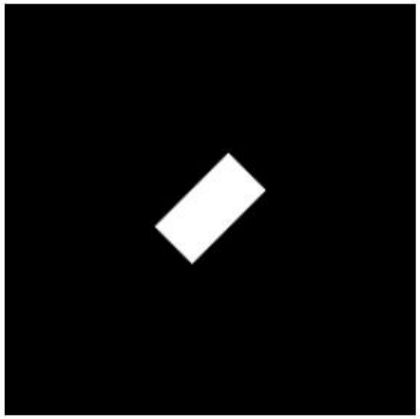
Examples of 2D DFT



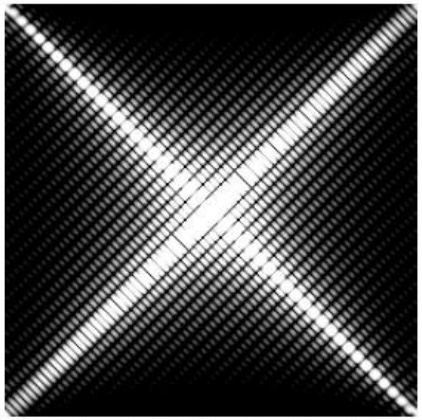
Test Image



log10 Fourier Spectrum



Rotated Test Image



Rotated log10 Fourier Spectrum

MATLAB Sample Code

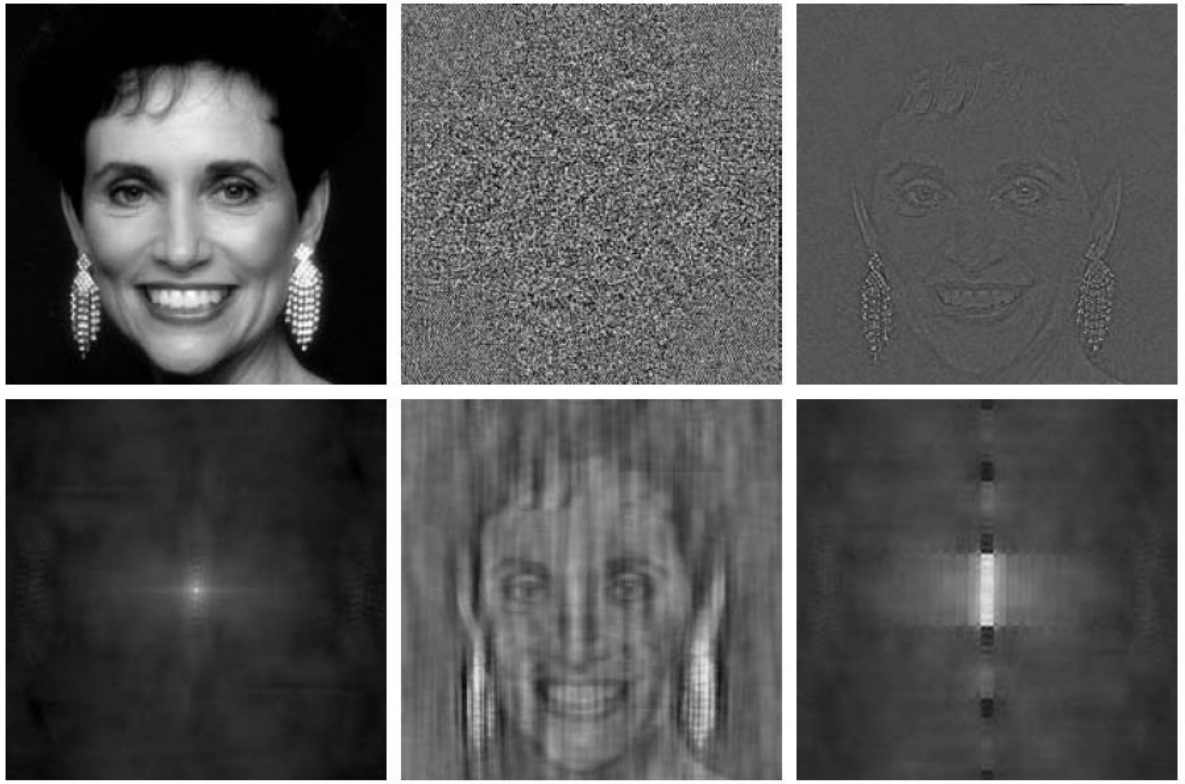
```
% Create test image and show it
img=zeros(256,256);
img(128-32:128+32,128-16:128+16)=1;
figure; imshow(img);
% Fourier Transform
IMG=fftshift(fft2(img));
% Show log10 of the Spectrum
figure; imshow(log10(1+abs(IMG)));
```

MATLAB Sample Code

```
% Rotate the test image
img45=imrotate(img,45,'bicubic','crop');
figure; imshow(img45);
% Fourier Transform
IMG45=fftshift(fft2(img45));
% Show log10 of the Spectrum
figure; imshow(log10(1+abs(IMG45)));
```

Examples of 2D DFT

- Combining the spectrum and the phase angle



a b c
d e f

FIGURE 4.27 (a) Woman. (b) Phase angle. (c) Woman reconstructed using only the phase angle. (d) Woman reconstructed using only the spectrum. (e) Reconstruction using the phase angle corresponding to the woman and the spectrum corresponding to the rectangle in Fig. 4.24(a). (f) Reconstruction using the phase of the rectangle and the spectrum of the woman.

Convolution

The ***convolution*** of two 1-dimensional functions $f(x)$ and $g(x)$ is generally denoted by $f(x) * g(x)$ and defined by the integral

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\alpha)g(x-\alpha)d\alpha \qquad (3.51)$$

where α is a dummy variable.

The importance of convolution in the frequency domain analysis lies in the fact that $f(x) * g(x)$ and $F(u)G(u)$ constitute a Fourier Transformation pair thus

$$f(x) * g(x) \quad \Leftrightarrow \quad F(u)G(u) \qquad (3.52)$$

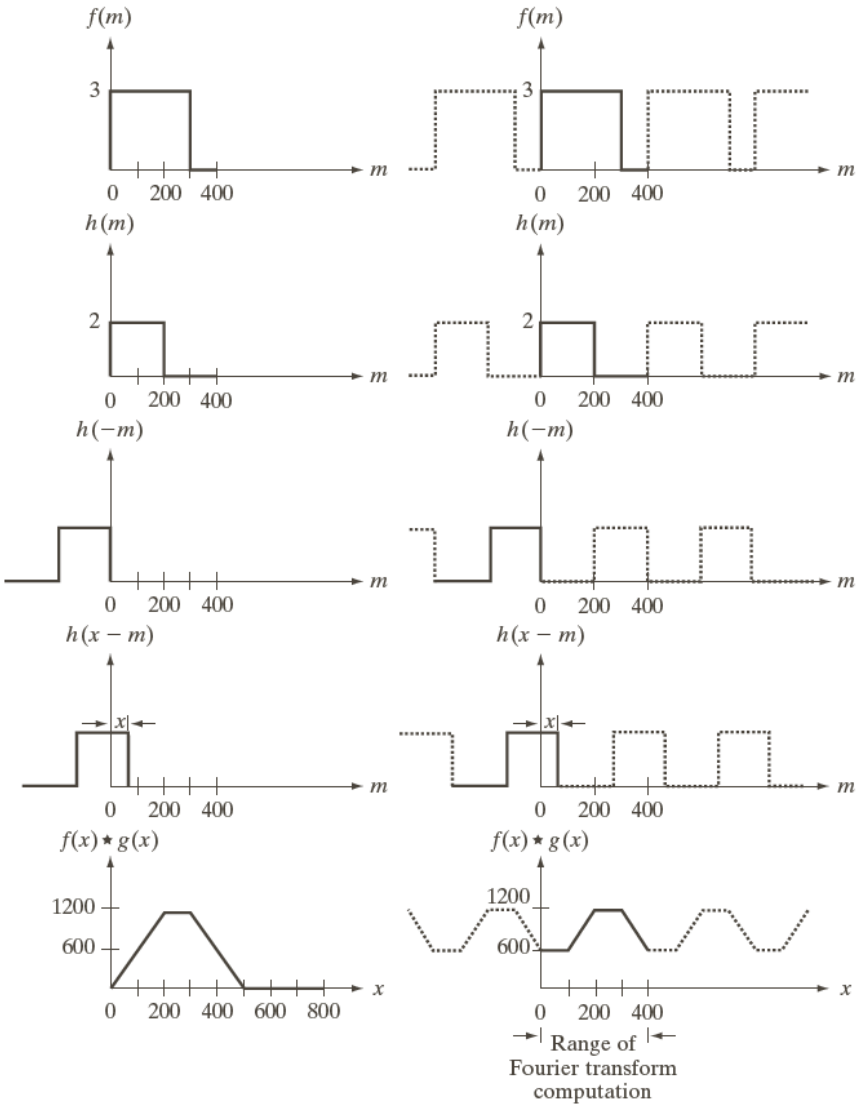
a convolution in the Image domain results in a multiplication in the Frequency domain, and vice versa

$$f(x)g(x) \quad \Leftrightarrow \quad F(u) * G(u) \qquad (3.53)$$

Convolution of a 1D signal

- Example on convolution

$$f(x) * h(x) = \sum_{m=0}^{399} f(x)h(x-m)$$



a f
b g
c h
d i
e j

FIGURE 4.28 Left column: convolution of two discrete functions obtained using the approach discussed in Section 3.4.2. The result in (e) is correct. Right column: Convolution of the same functions, but taking into account the periodicity implied by the DFT. Note in (j) how data from adjacent periods produce wraparound error, yielding an incorrect convolution result. To obtain the correct result, function padding must be used.

Summary of DFT definitions

Name	Expression(s)
1) Discrete Fourier transform (DFT) of $f(x, y)$	$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$
2) Inverse discrete Fourier transform (IDFT) of $F(u, v)$	$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)}$
3) Polar representation	$F(u, v) = \lvert F(u, v) \rvert e^{j\phi(u,v)}$
4) Spectrum	$\lvert F(u, v) \rvert = \left[R^2(u, v) + I^2(u, v) \right]^{1/2}$ $R = \text{Real}(F); \quad I = \text{Imag}(F)$
5) Phase angle	$\phi(u, v) = \tan^{-1}\left[\frac{I(u, v)}{R(u, v)} \right]$
6) Power spectrum	$P(u, v) = \lvert F(u, v) \rvert^2$
7) Average value	$\bar{f}(x, y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) = \frac{1}{MN} F(0, 0)$

(Continued)

Name	Expression(s)
8) Periodicity (k_1 and k_2 are integers)	$F(u, v) = F(u + k_1 M, v) = F(u, v + k_2 N)$ $= F(u + k_1 M, v + k_2 N)$ $f(x, y) = f(x + k_1 M, y) = f(x, y + k_2 N)$ $= f(x + k_1 M, y + k_2 N)$
9) Convolution	$f(x, y) \star h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x - m, y - n)$
10) Correlation	$f(x, y) \star h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n) h(x + m, y + n)$
11) Separability	The 2-D DFT can be computed by computing 1-D DFT transforms along the rows (columns) of the image, followed by 1-D transforms along the columns (rows) of the result. See Section 4.11.1.
12) Obtaining the inverse Fourier transform using a forward transform algorithm.	$MNf^*(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F^*(u, v) e^{-j2\pi(ux/M + vy/N)}$ This equation indicates that inputting $F^*(u, v)$ into an algorithm that computes the forward transform (right side of above equation) yields $MNf^*(x, y)$. Taking the complex conjugate and dividing by MN gives the desired inverse. See Section 4.11.2.

Summary of DFT pairs

Name	DFT Pairs
1) Symmetry properties	See Table 4.1
2) Linearity	$af_1(x, y) + bf_2(x, y) \Leftrightarrow aF_1(u, v) + bF_2(u, v)$
3) Translation (general)	$f(x, y)e^{j2\pi(u_0x/M + v_0y/N)} \Leftrightarrow F(u - u_0, v - v_0)$ $f(x - x_0, y - y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M + vy_0/N)}$
4) Translation to center of the frequency rectangle, $(M/2, N/2)$	$f(x, y)(-1)^{x+y} \Leftrightarrow F(u - M/2, v - N/2)$ $f(x - M/2, y - N/2) \Leftrightarrow F(u, v)(-1)^{u+v}$
5) Rotation	$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0)$ $x = r\cos\theta \quad y = r\sin\theta \quad u = \omega\cos\varphi \quad v = \omega\sin\varphi$
6) Convolution theorem†	$f(x, y) \star h(x, y) \Leftrightarrow F(u, v)H(u, v)$ $f(x, y)h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$

(*Continued*)

Name	DFT Pairs
7) Correlation theorem†	$f(x, y) \star h(x, y) \Leftrightarrow F^*(u, v)H(u, v)$ $f^*(x, y)h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$
8) Discrete unit impulse	$\delta(x, y) \Leftrightarrow 1$
9) Rectangle	$\text{rect}[a, b] \Leftrightarrow ab\frac{\sin(\pi ua)}{(\pi ua)}\frac{\sin(\pi vb)}{(\pi vb)}e^{-j\pi(ua+vb)}$
10) Sine	$\sin(2\pi u_0x + 2\pi v_0y) \Leftrightarrow$ $j\frac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) - \delta(u - Mu_0, v - Nv_0)\right]$
11) Cosine	$\cos(2\pi u_0x + 2\pi v_0y) \Leftrightarrow$ $\frac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) + \delta(u - Mu_0, v - Nv_0)\right]$

The following Fourier transform pairs are derivable only for continuous variables, denoted as before by t and z for spatial variables and by μ and ν for frequency variables. These results can be used for DFT work by sampling the continuous forms.

12) *Differentiation* (The expressions on the right assume that $f(\pm\infty, \pm\infty) = 0$.)	$\left(\frac{\partial}{\partial t}\right)^m\left(\frac{\partial}{\partial z}\right)^n f(t, z) \Leftrightarrow (j2\pi\mu)^m(j2\pi\nu)^n F(\mu, \nu)$ $\frac{\partial^m f(t, z)}{\partial t^m} \Leftrightarrow (j2\pi\mu)^m F(\mu, \nu); \frac{\partial^n f(t, z)}{\partial z^n} \Leftrightarrow (j2\pi\nu)^n F(\mu, \nu)$
13) *Gaussian*	$A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)} \Leftrightarrow Ae^{-(\mu^2+\nu^2)/2\sigma^2}$ (A is a constant)

†Assumes that the functions have been extended by zero padding. Convolution and correlation are associative, commutative, and distributive.

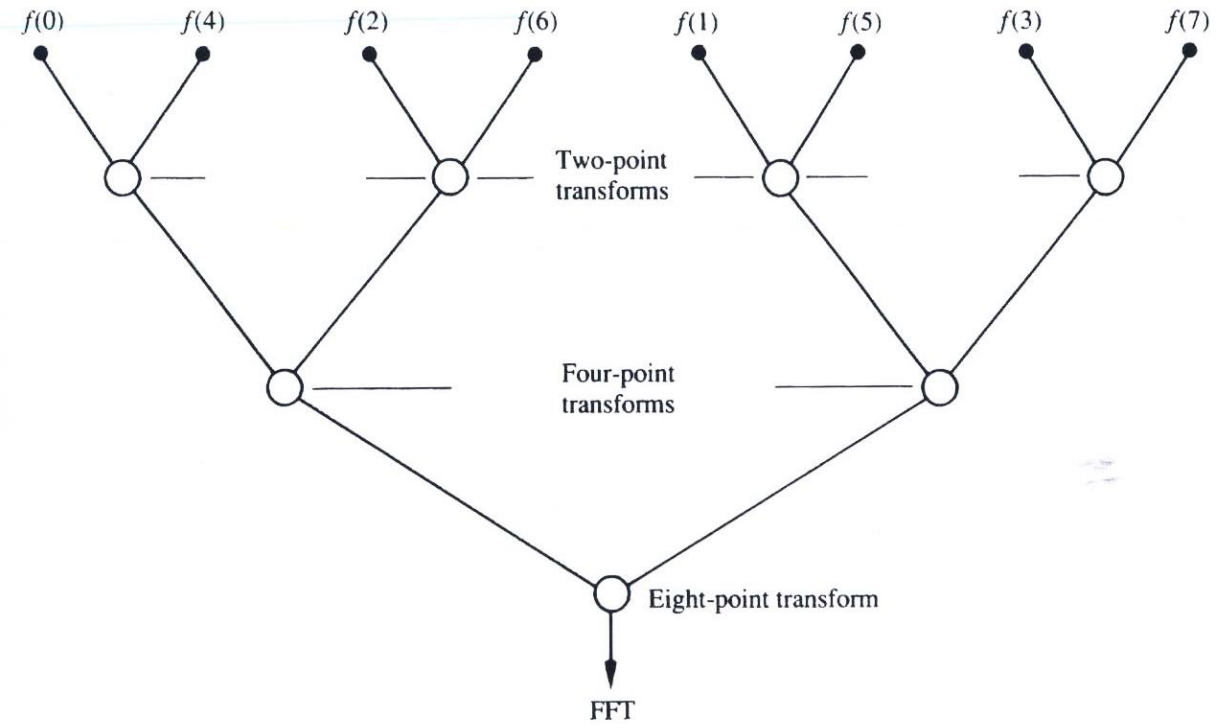
Fast Fourier Transform (FFT)

- Brute-force implementation of Fourier Transform requires on the order of $(MN)^2$ summations and additions
- 1024 x 1024 pixels

  => the order of a trillion multiplications and summations for one DFT

  => computationally too heavy
- FFT => the order of $MN\ log_2 MN$
- Based on the successive doubling method

Fast Fourier Transform (FFT)



Fast Fourier Transform (FFT)

Table 3.1 A Comparison of N^2 versus $N \log_2 N$ for Various Values of N

N	N^2 (*Direct* FT)	$N \log_2 N$ (FFT)	*Computational Advantage* ($N/\log_2 N$)
2	4	2	2.00
4	16	8	2.00
8	64	24	2.67
16	256	64	4.00
32	1,024	160	6.40
64	4,096	384	10.67
128	16,384	896	18.29
256	65,536	2,048	32.00
512	262,144	4,608	56.89
1024	1,048,576	10,240	102.40
2048	4,194,304	22,528	186.18
4096	16,777,216	49,152	341.33
8192	67,108,864	106,496	630.15

DISCRETE FAST FOURIER TRANSFORM (FFT)

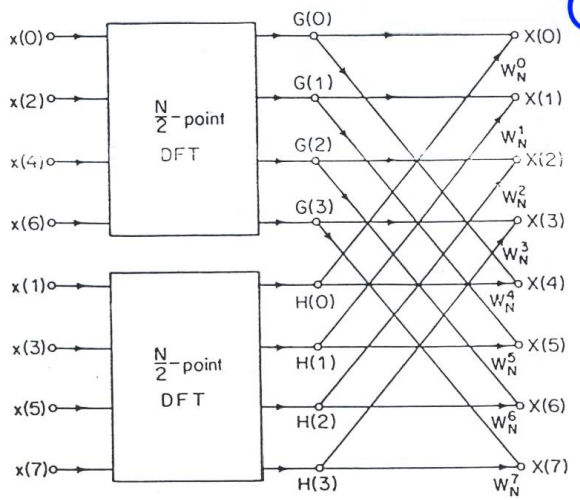


Fig. 6.3 Flow graph of the decimation-in-time decomposition of an N-point DFT computation into two $N/2$-point DFT computations ($N = 8$).

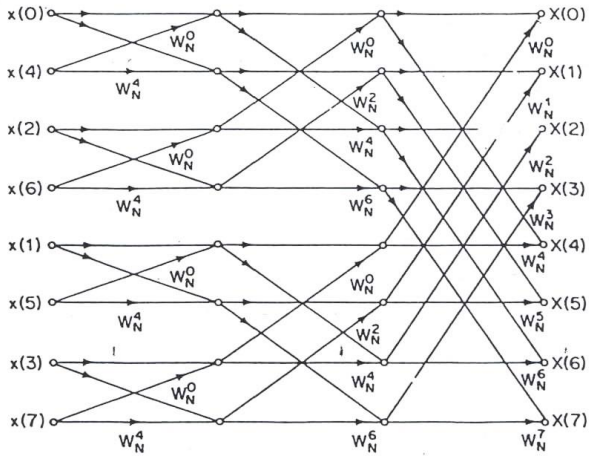


Fig. 6.7 Flow graph of complete decimation-in-time decomp sition of an eight-point DFT computation.

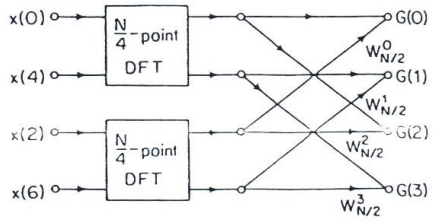


Fig. 6.4 Flow graph of the decimation-in-time decomposition of an $N/2$-point DFT computation into two $N/4$-point DFT computations ($N = 8$).

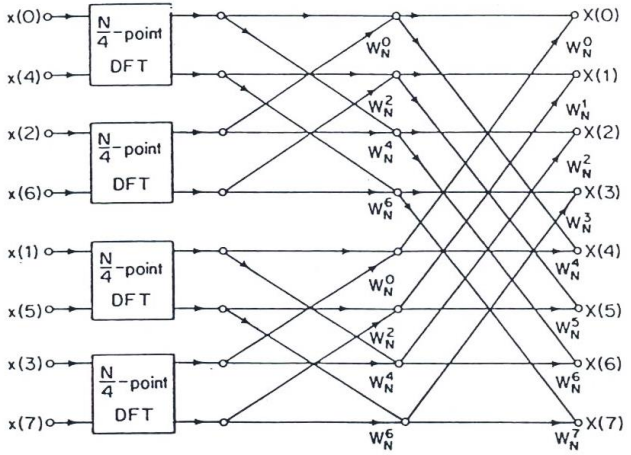


Fig. 6.5 Result of substituting Fig. 6.4 into Fig. 6.3.

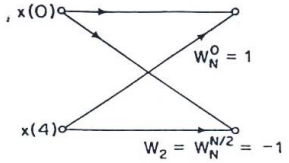
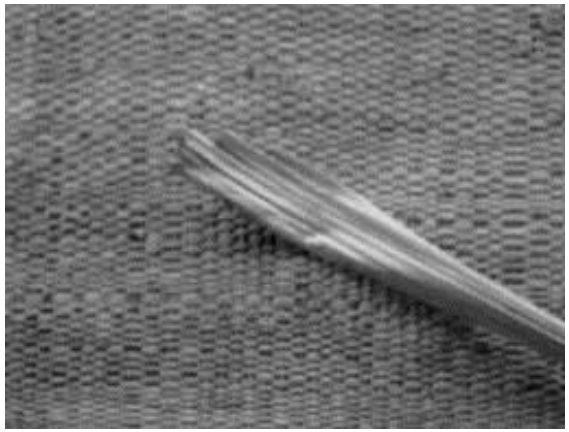


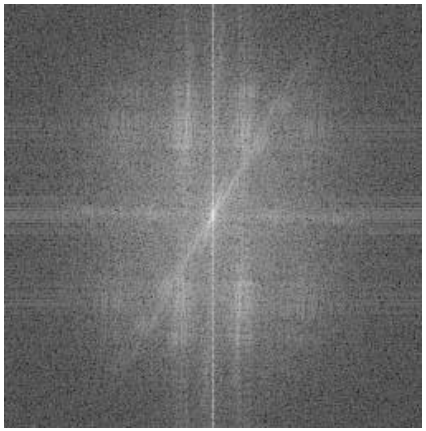
Fig. 6.6 Flow graph of a two-point DFT.

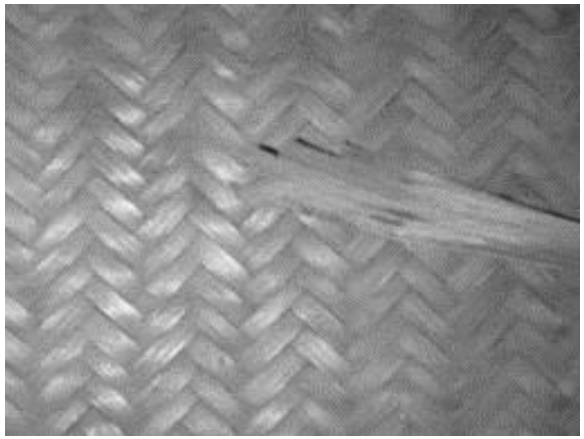
FT for Image Proc. and Analysis

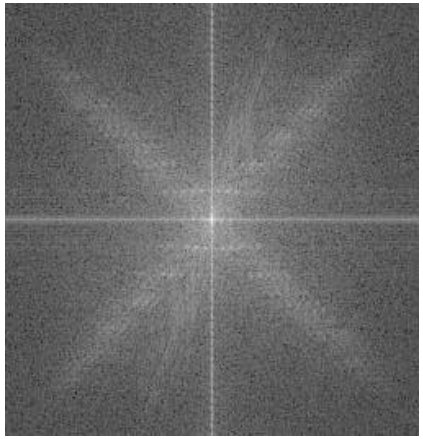
- Properties of Fourier Transform offer for example the following use cases:
  - Feature extraction
    - Frequency domain features
  - Image compression
  - Image filtering
    - Image enhancement in the frequency domain
    - Preprocessing

FT: Feature Extraction







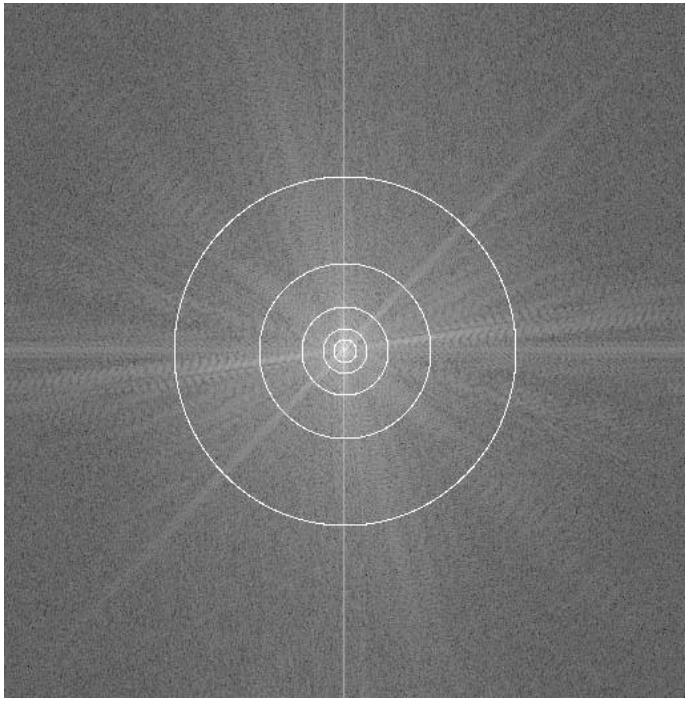


- Fourier transforms of texture
- Regular patterns (and irregularities in regular patterns)
- Feature extraction

FT: Image Compression

- Not so many coefficients needed => image compression.
- Lossy vs. lossless compression?

Radius (pixels)	% Image power
8	95
16	97
32	98
64	99.4
128	99.8

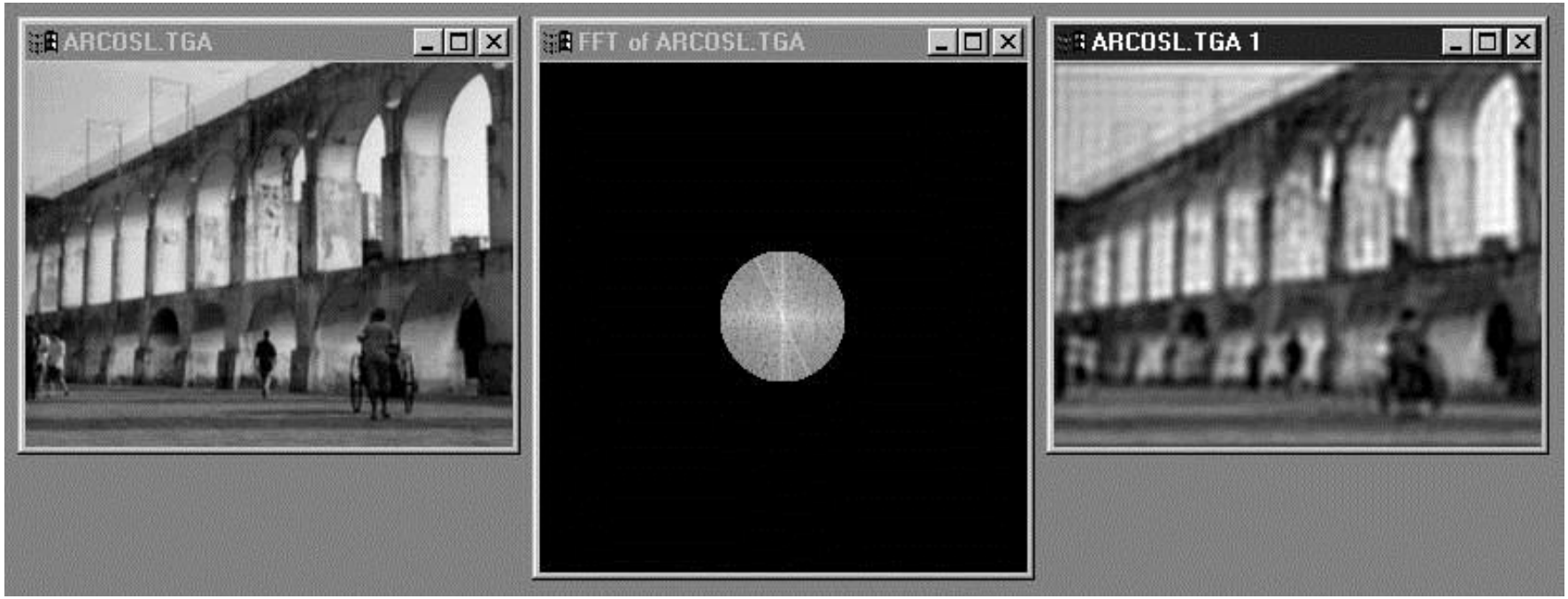


FT: Image Filtering

- Enhancement in the frequency domain:

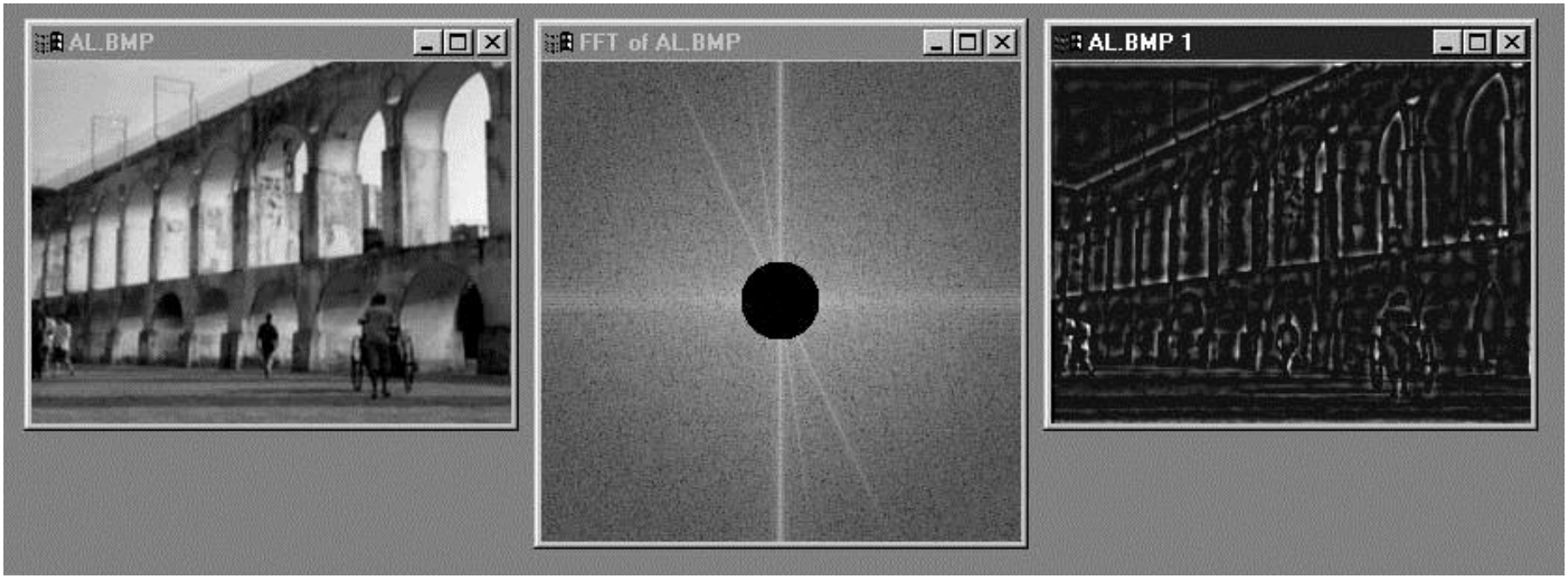
$$f(x) * g(x) \leftrightarrow F(u)G(u)$$

- Original, ideal low pass filter, and filtered image (l-c-r)



FT: Image Filtering

- Ideal high pass filter:
  - Original (left) and filtered image (right).



Gabor Filtering

- For local and global feature extraction.
- Filtering in time (spatial) space and frequency space.
- For image processing and analysis two important parameters:
  - Frequency f and orientation θ.
- Application example: Face detection and recognition.
  - FACEDETECT project, two partners:
    - MVPR, LUT, Finland, and
    - Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, UK.

Feature Detector: 2-D Gabor Filter

- Transform ψ of a time domain signal in (x, y)

$$\psi(x,y) = \frac{f^2}{\pi\gamma\eta} e^{-\left(\frac{f^2}{\gamma^2}x'^2 + \frac{f^2}{\eta^2}y'^2\right)} e^{j2\pi f x'}$$

$$x' = x\cos\theta + y\sin\theta$$

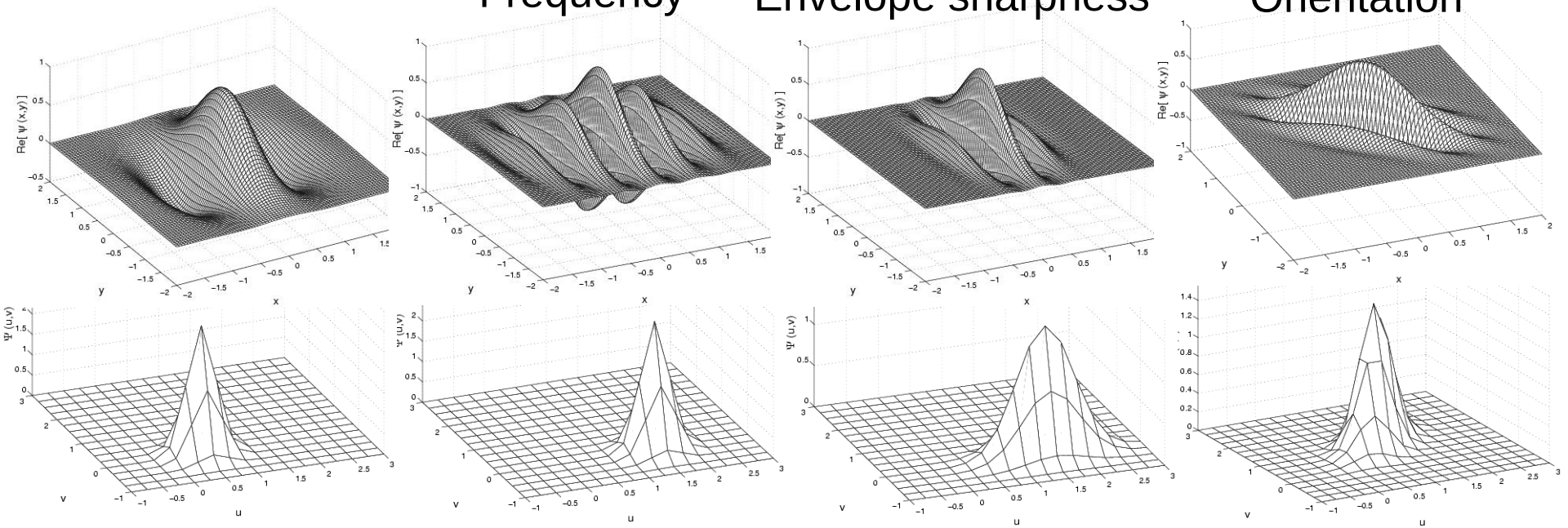
$$y' = -x\sin\theta + y\cos\theta$$

- i.e, modulating a Gaussian with a sinusoidal function

Gabor Features

- Maximal joint localization in the spatial and frequency domain, smooth and noise tolerant
- Parameters for invariance manipulation

Frequency Envelope sharpness Orientation



Constructing Response Matrix

- Filter response $r_\xi(x,y; f,\theta)$ can be calculated for various frequencies f and orientations θ to construct a response matrix.

$$\mathbf{R} = \begin{bmatrix} r_\xi(x_0,y_0;f_1,\theta_1) & \cdots & r_\xi(x_0,y_0;f_1,\theta_{N-1}) & r_\xi(x_0,y_0;f_1,\theta_0) \\ \vdots & & \vdots & \vdots \\ r_\xi(x_0,y_0;f_{M-1},\theta_1) & \cdots & r_\xi(x_0,y_0;f_{M-1},\theta_{N-1}) & r_\xi(x_0,y_0;f_{M-1},\theta_0) \end{bmatrix}$$

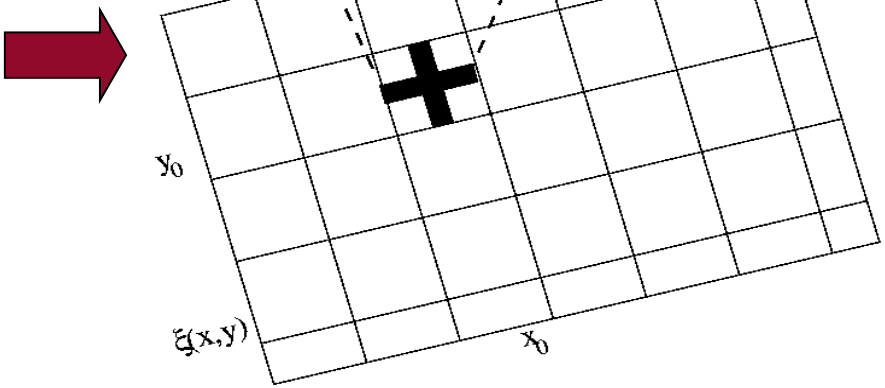


Image scaling appears as a shift of the rows (high frequencies may vanish)

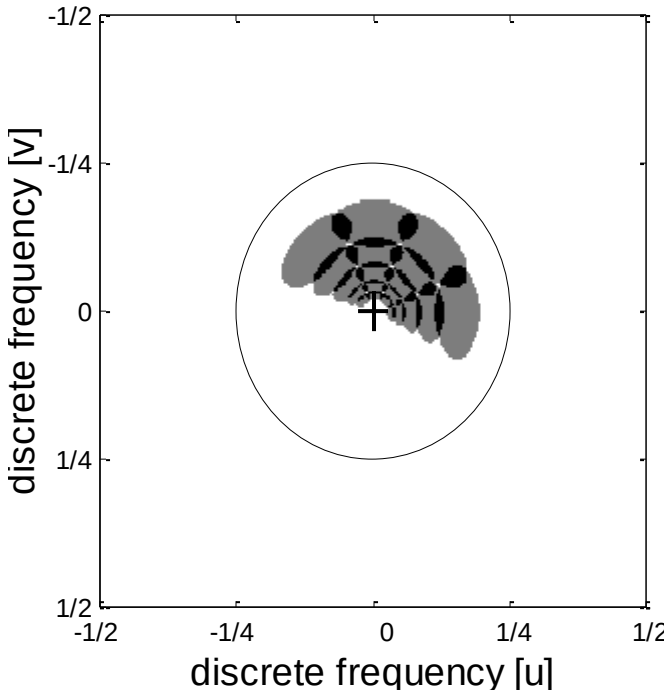
Image rotation appears as a circular shift of the columns.

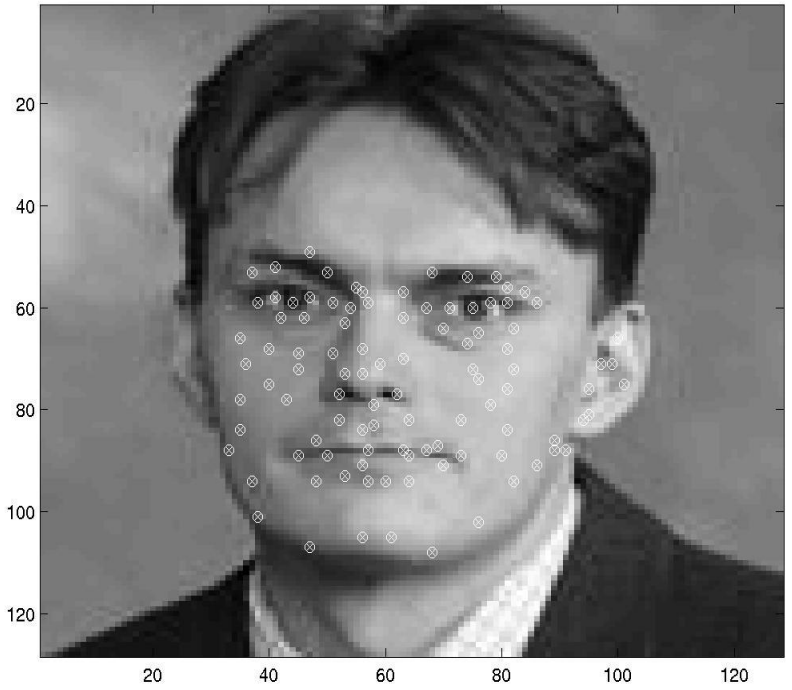
Columns represent orientations, rows represent frequencies.

A SCALE AND ROTATION INVARIANT TREATMENT OF THE RESPONSE MATRIX CAN BE ESTABLISHED, AND THUS, WE CAN CONCENTRATE ONLY HOW TO CLASSIFY THEM IN THE STANDARD POSE.

2-D Gabor Features

- What do the filters ”se



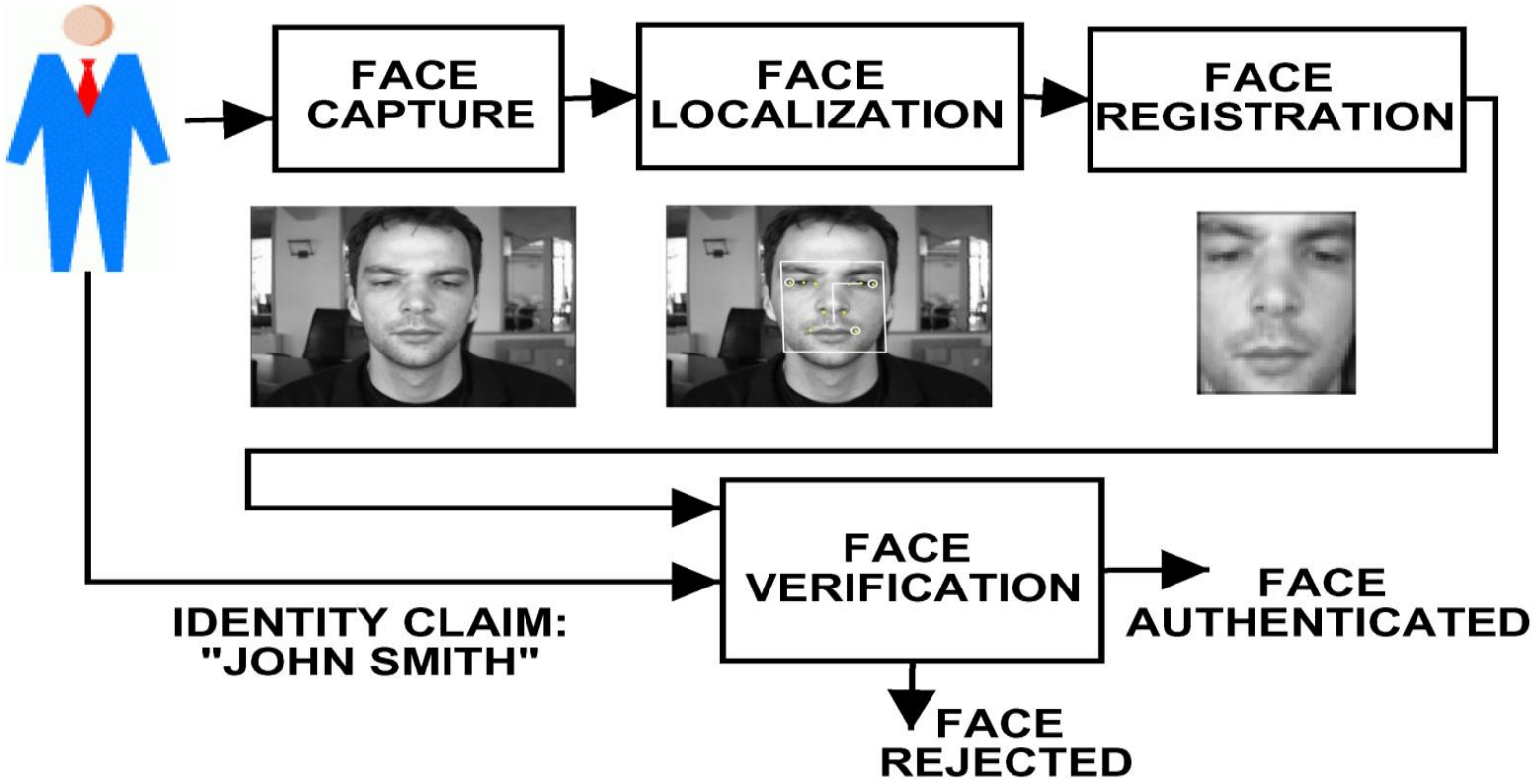


Face Detection and Recognition

- **Face verification (authentication)**
  Validating a claimed identity based on the image of a face: Are you Mr./Ms. X?
- **Face recognition (identification)**
  Identifying a person based on an image of his/her face: Who are you?
- **Face detection/localization**
  Location of human faces in images at different positions, scales, orientations, and lighting conditions.

Face Detection and Recognition

- Example on authentication based on facial information



Face Det/Rec.:Proposed Algorithm

- Avoiding a scanning window.
- Using feature detectors.
- Shape-free texture model for the final decision.

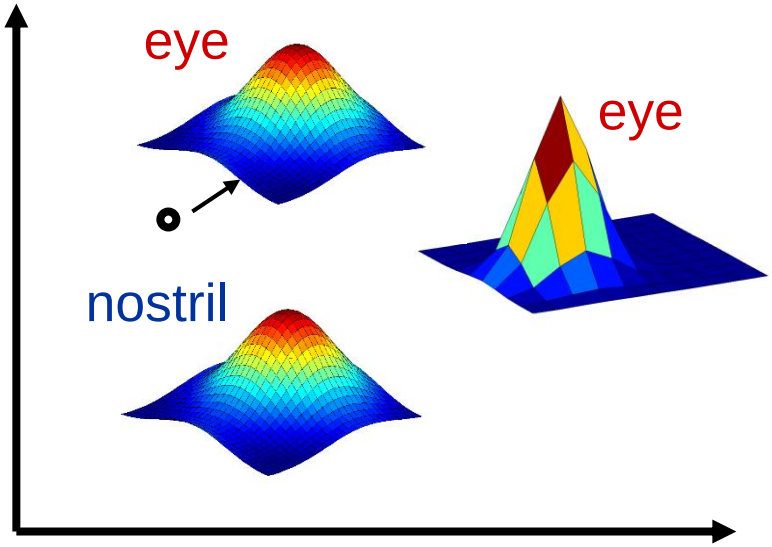
Face Det/Rec.:Evidence Extraction

- Requirements
  - Scale invariant extraction.
  - Rotation invariant extraction.
  - Provides sufficiently small amount of correct candidate points (n best points from each class; needs confidence measure).
- Preferred
  - Estimation of evidence scale and orientation.
  - Fast extraction (scalability).

Face Det/Rec.: Classifier Construction

- Stability property guarantees approximately the Gaussian form of classes in the feature space.
- One class may still consist of several sub-clusters (open eye, closed eye, etc.).

Bayesian classification of features

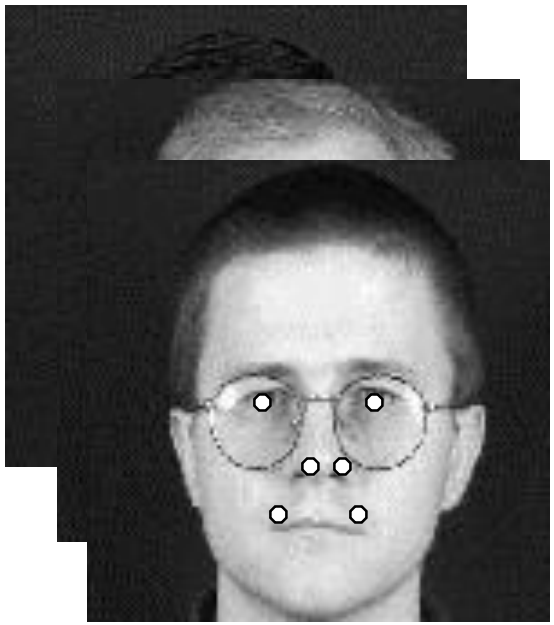


Gaussian mixture model densities (EM estimation)

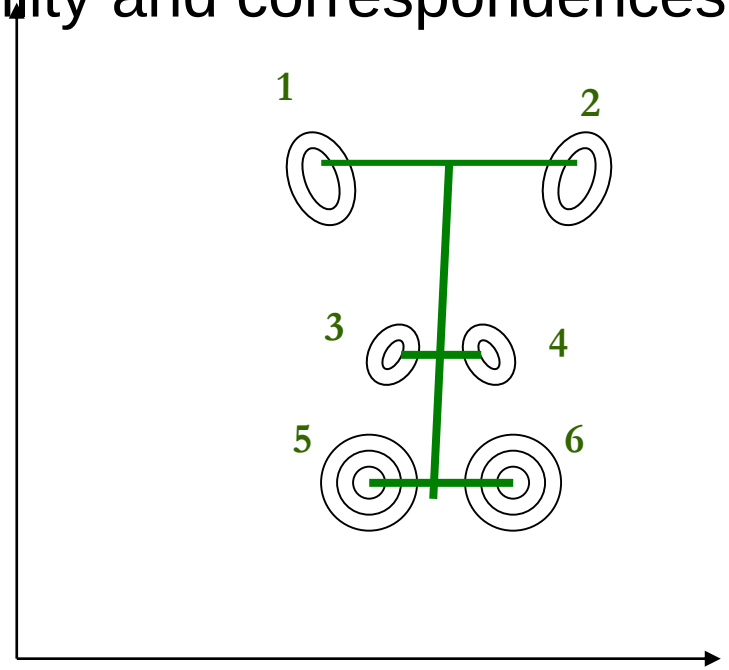
Face Det./Rec.: Affine Learned Correspondences

Aligned images of objects

and manually selected features



Variability and correspondences

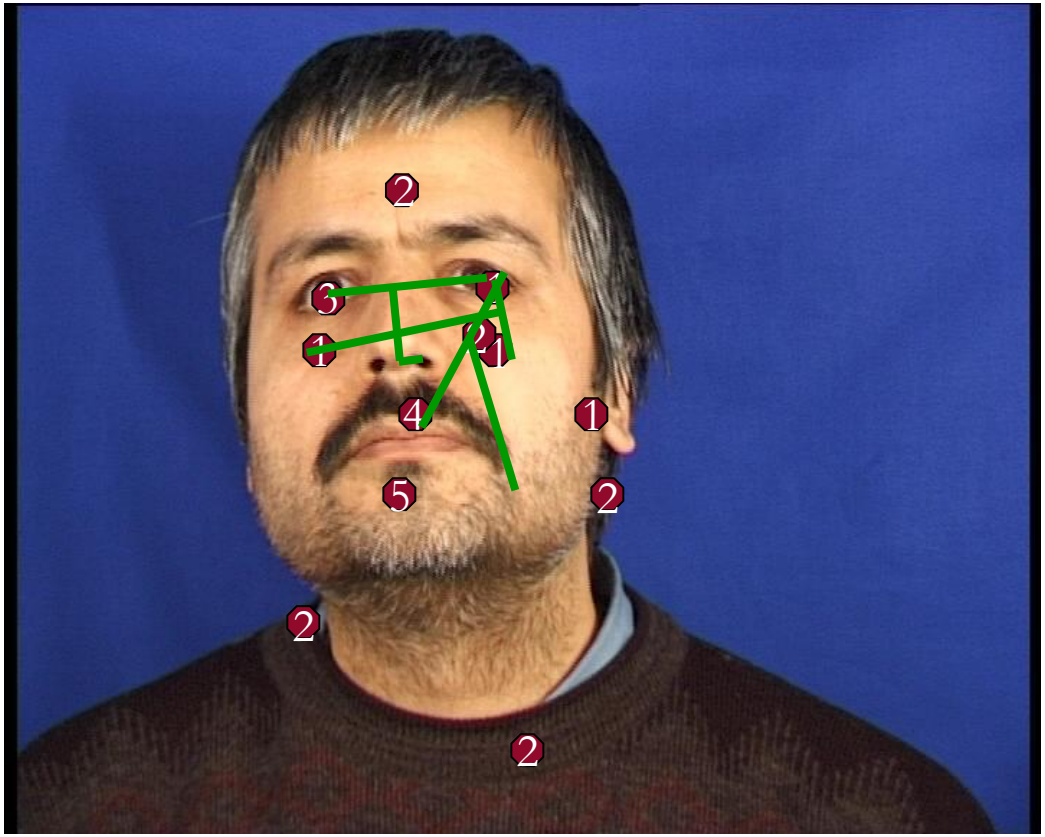


Face Det./Rec.: Affine Learned Correspondences

1. Evidence extraction.

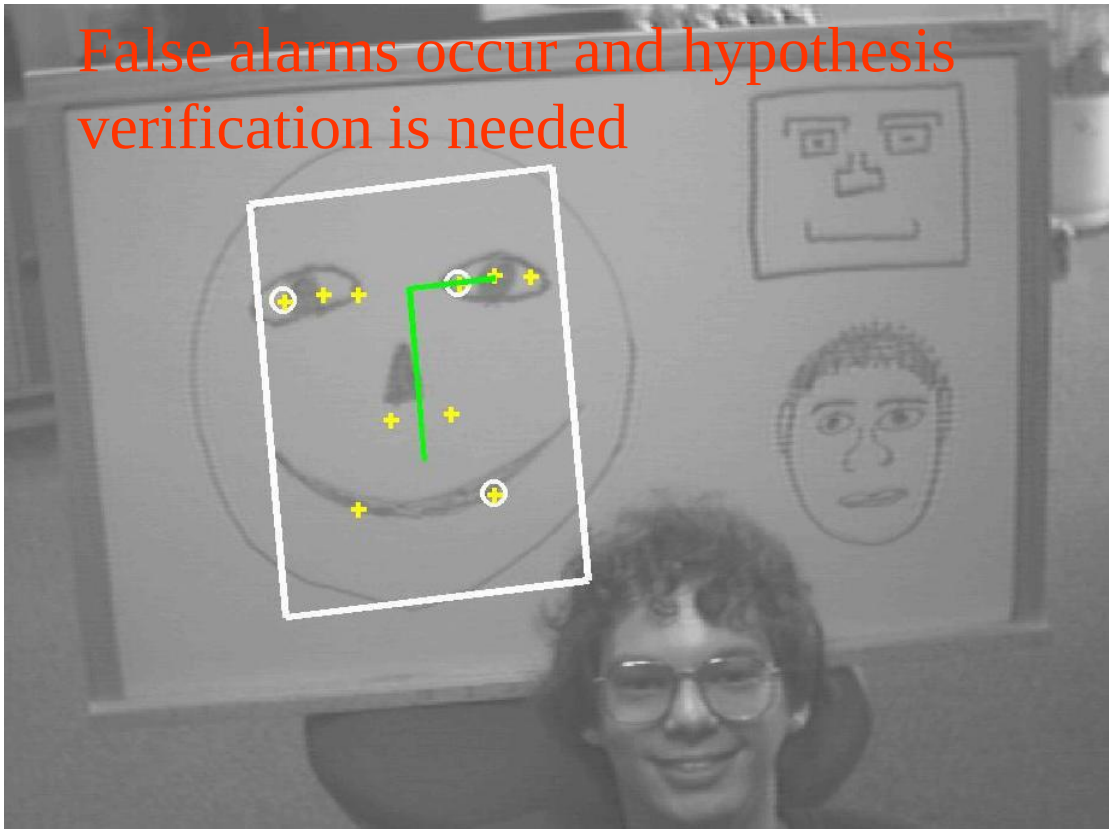
Instance approved

2. Affine search and match to correspondence model.



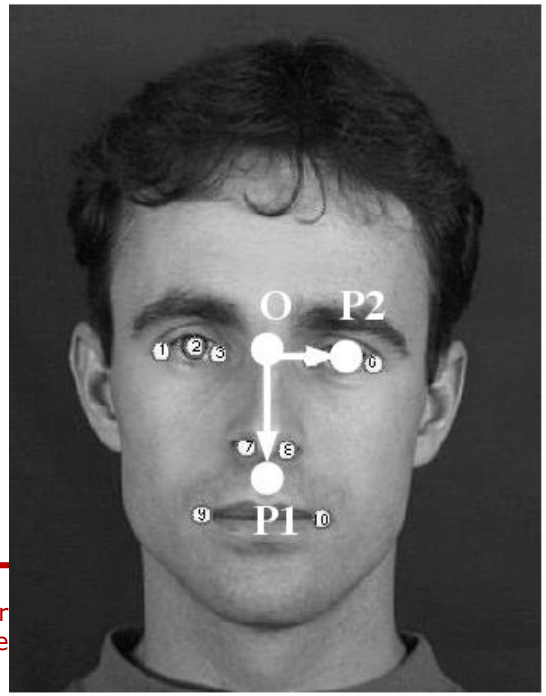
Face Det./Rec.: False alarms

- False alarms occur and hypothesis verification is needed



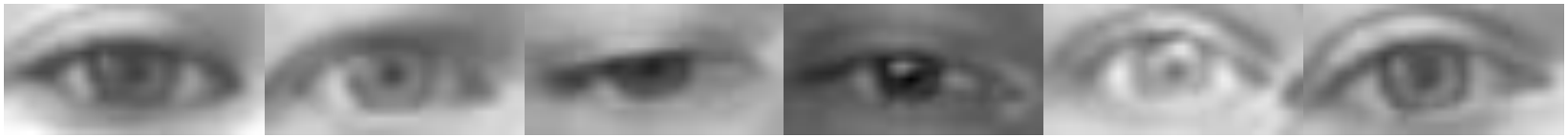
Face Det./Rec.: Face Space

- Normalization of space where shape variations and capture effects are removed from patterns
- Based on three points on the face -> affine registration
- Optimal with regard to the photometric variance over a large set of faces

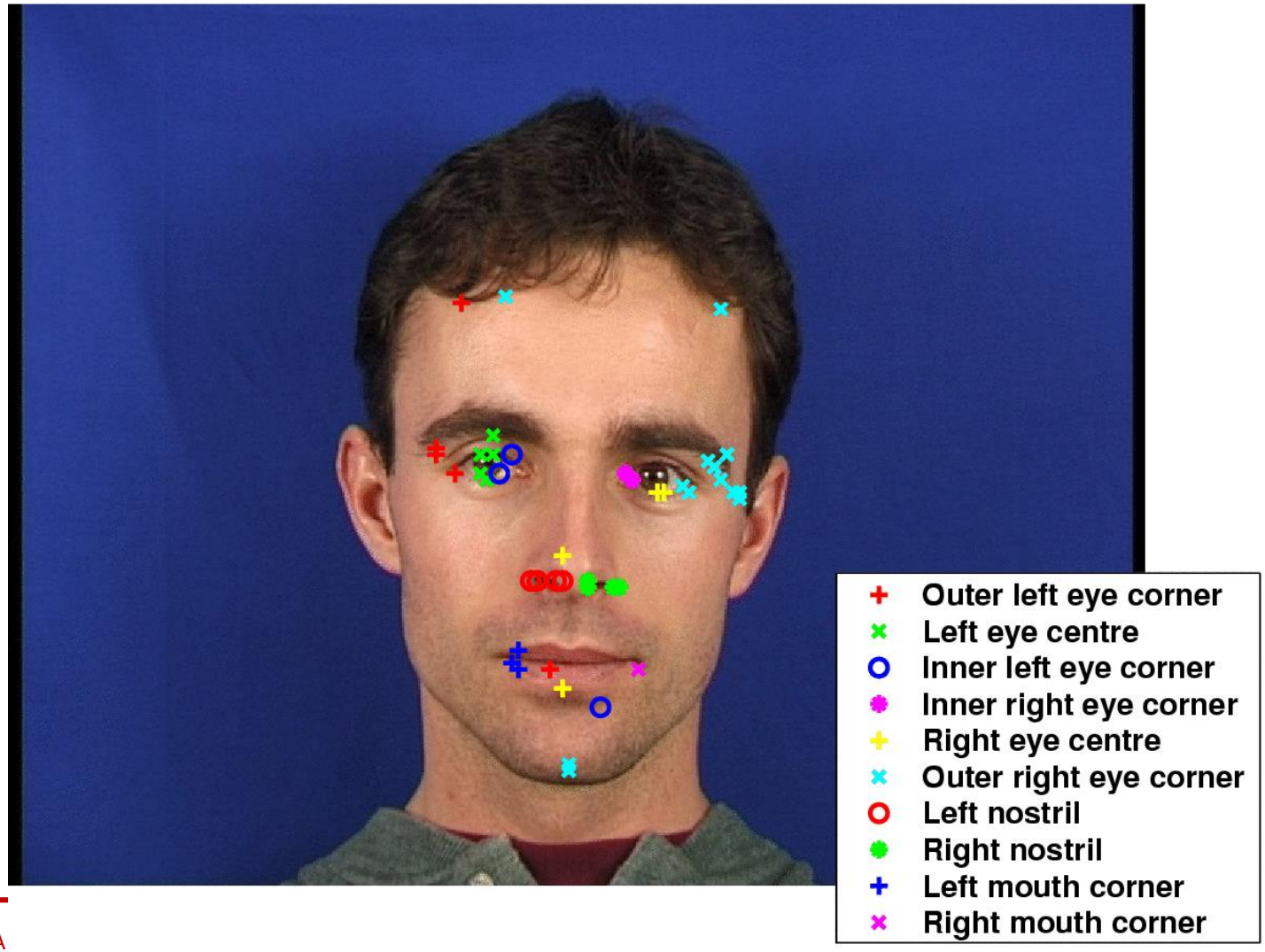


Face Det./Rec.: Features & Feature Detectors

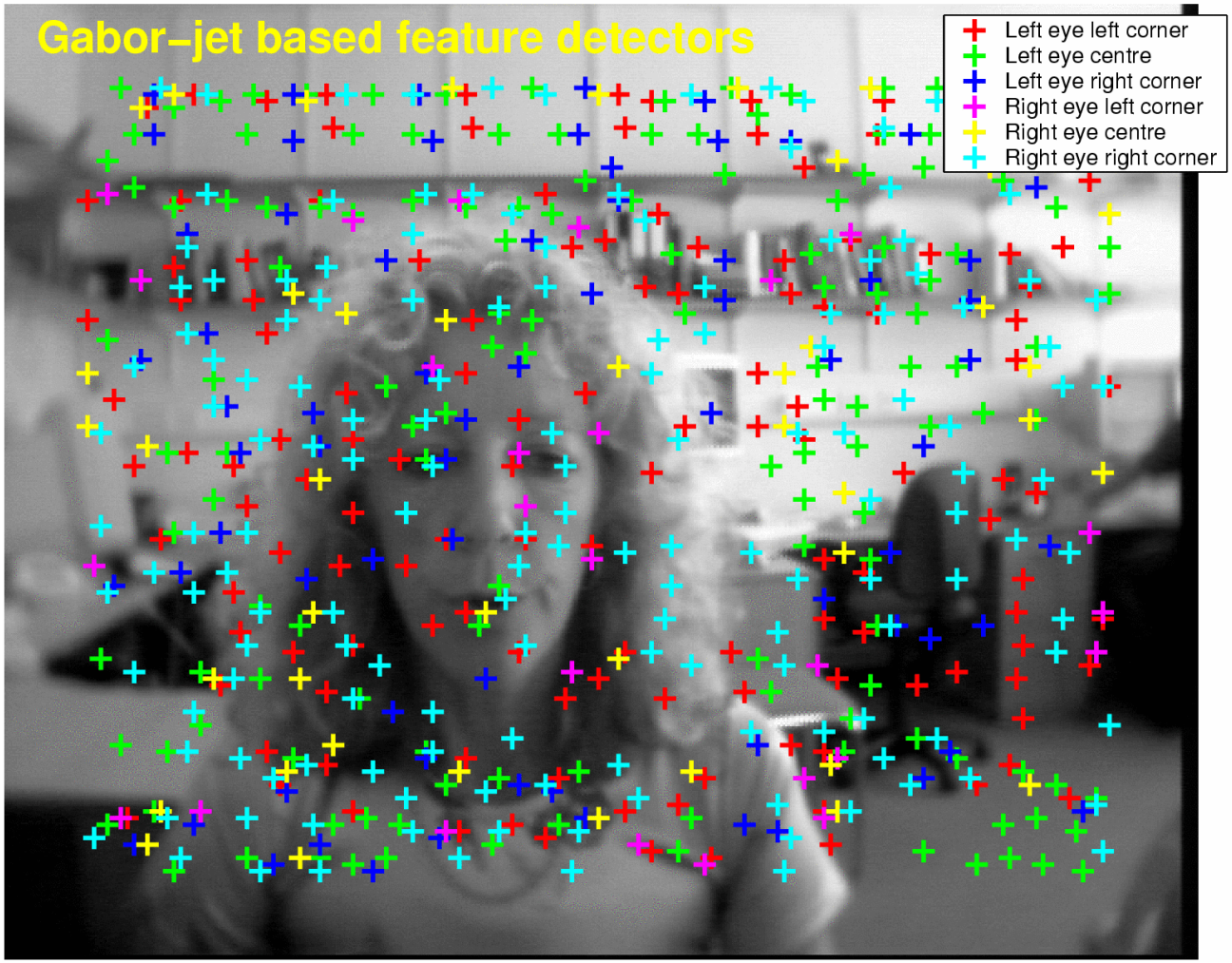
- Features = salient parts of face
- Small localization variance and frequent occurrence over population
- Illumination, scale, rotation, and translation invariance
- Automatic analysis using the face space desirable



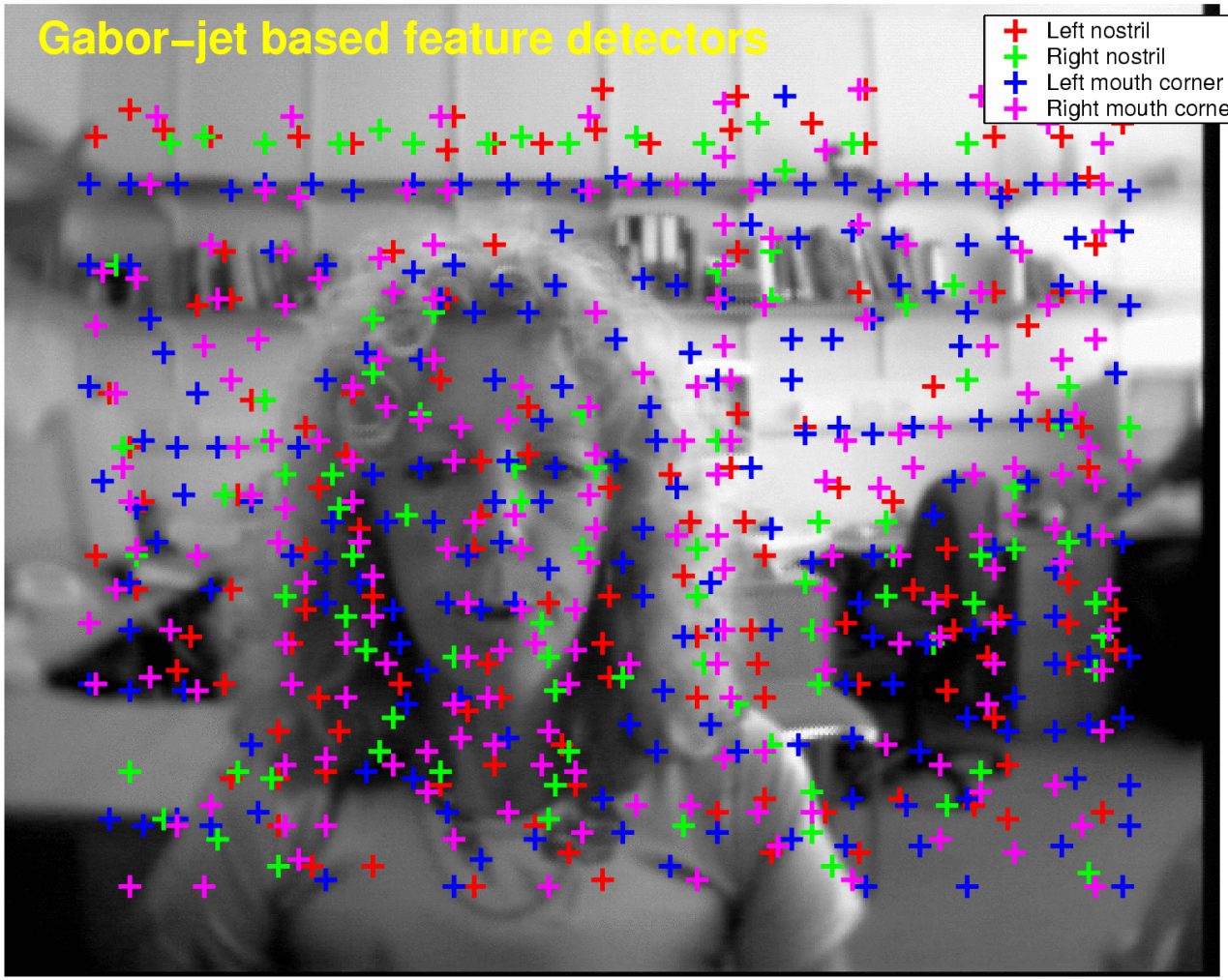
Face Det./Rec.: Features & Feature Detectors



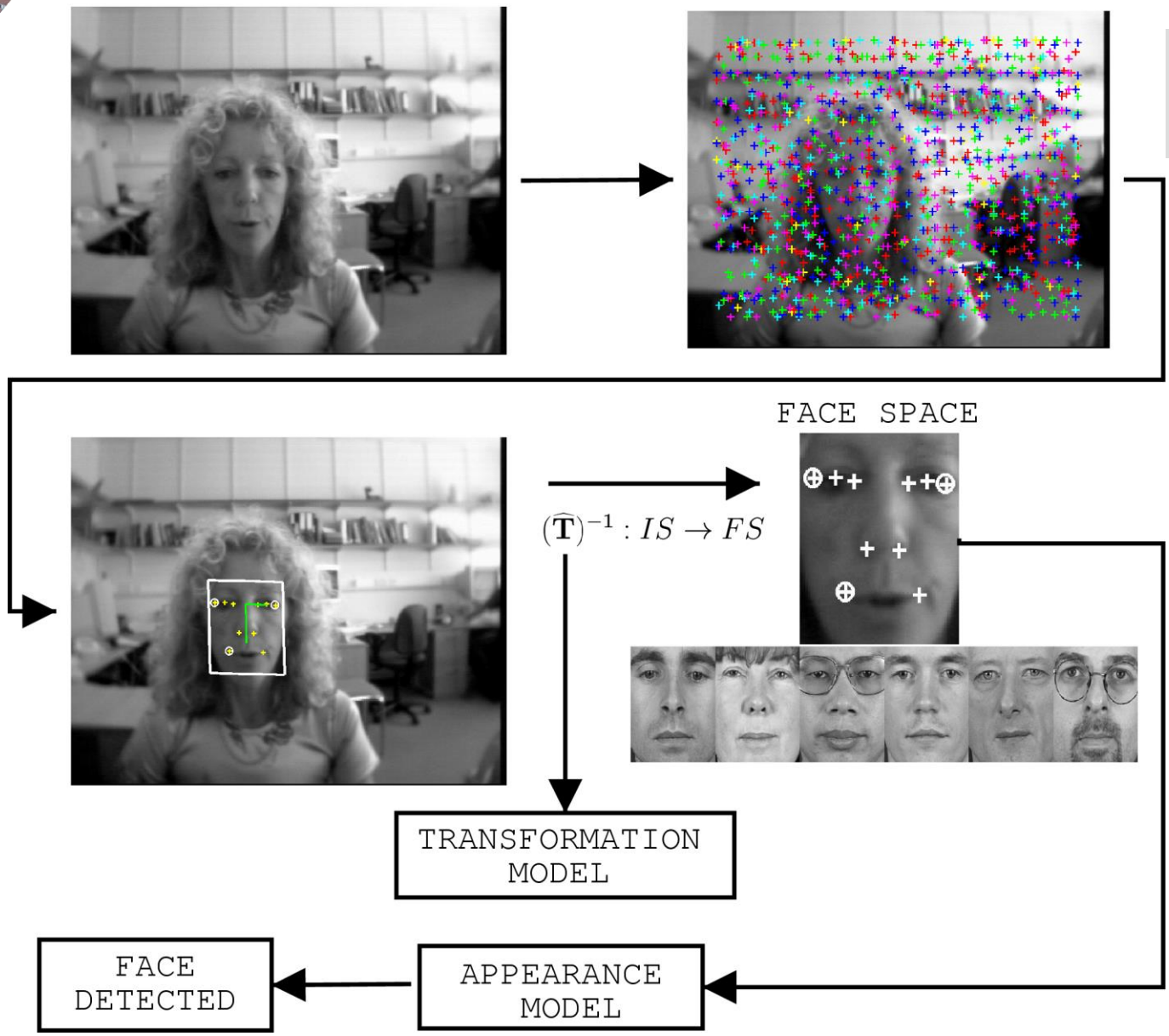
Face Det./Rec.: Features & Feature Detectors



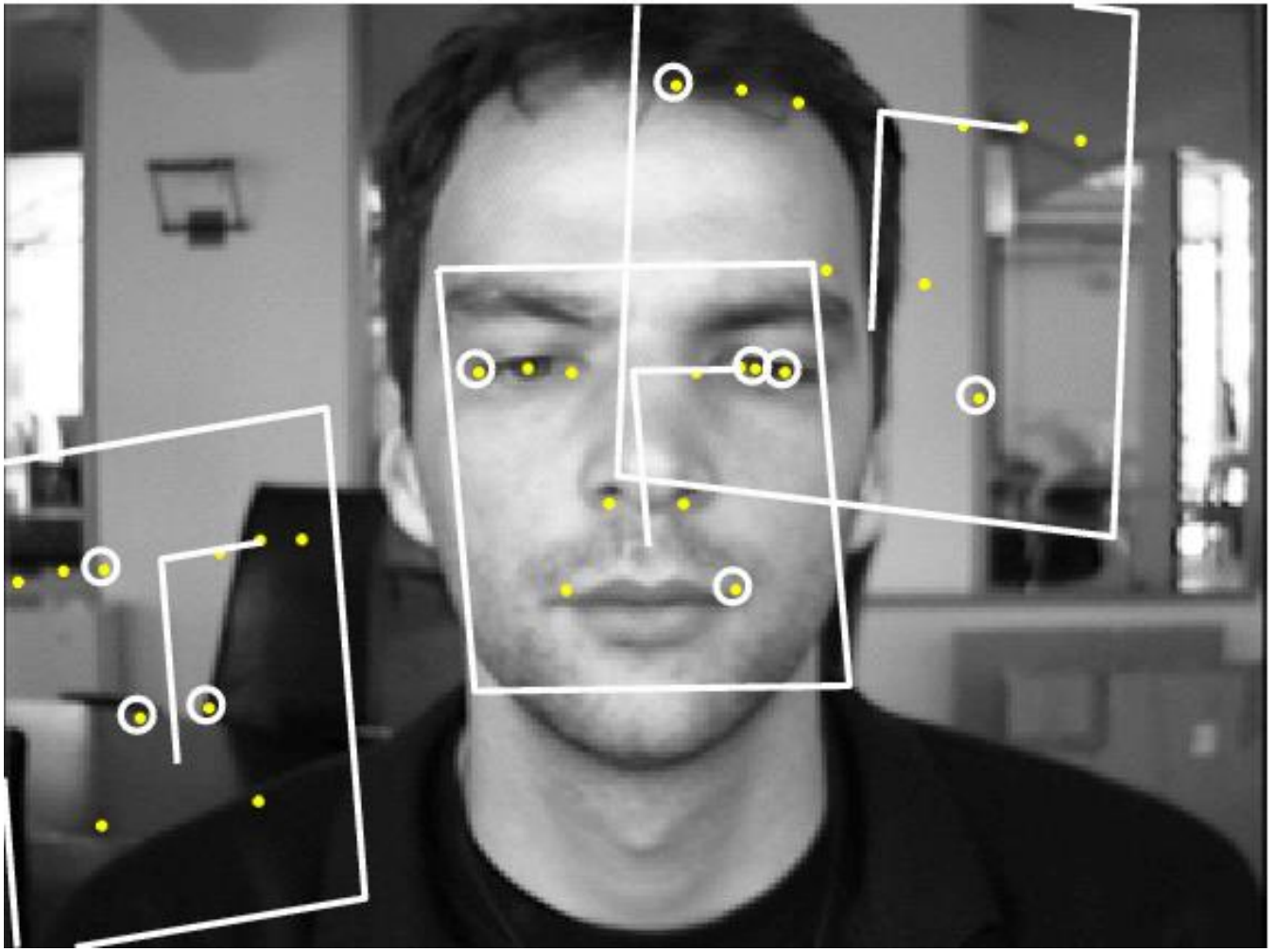
Face Det./Rec.: Features & Feature Detectors



- Outline of the prosess



Face Det./Rec.: Features & Feature Detectors

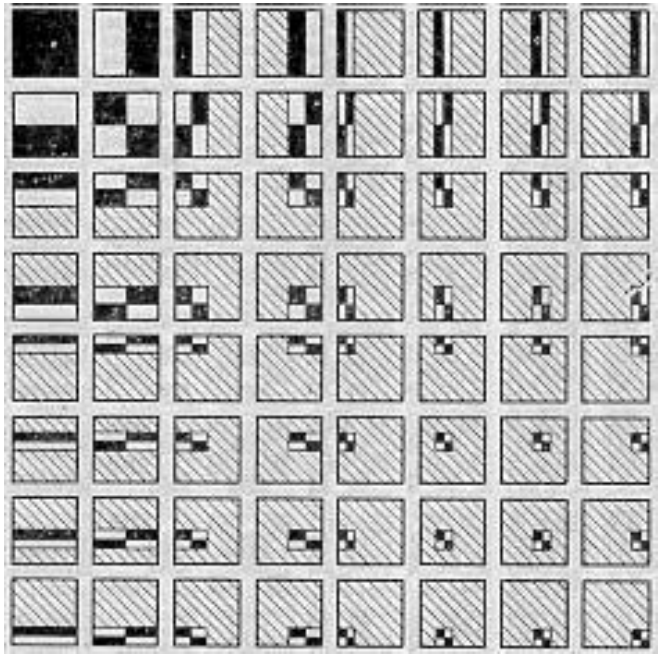


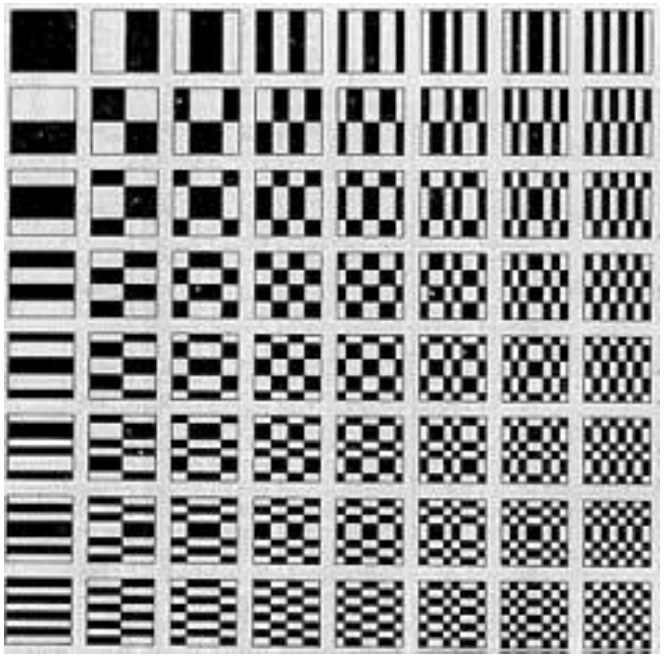
Other Transforms

- DFT: part of JPEG
  - Cosine, Sine
- Hadamar, Haar (square wave), Slant (sawtooth wave)
- Karhunen-Loeve, Fast KL: variance reconstruction
  - One component in the compression of spectral images
- SVD transform
- Wavelets
  - better energy compaction that with DFT
  - Similar applications as for DFT (e.g. JPEG2000)

Other Transforms

- Basis images: Haar (wavelets) (left), Hadamard (right).
  - Haar values: positive-black, negative-white, zero-gray
  - Hadamard values: one – black, minus one – white.





Summary

- Image transforms from the spatial domain into the frequency domain.
  - Fourier Transform.
  - For overall image processing and analysis.
  - Feature extraction, image compression, and image filtering (enhancement).
  - Discrete Fourier Transform (DFT) and Fast Fourier Transform (FFT).
  - Properties of 2-D Fourier Transform.
    - Separability, periodicity, translation, rotation, scaling, convolution, correlation.
- Gabor filtering.
  - For local and global feature extraction.
  - Orientation and frequency.
  - 2-D Gabor filter.
- Other transforms.
  - For example, Cosine Transform for image compression.
  - More detailed information later in the course.