Lecture 3 Image Sampling

COMP3204 Computer Vision

How is an image sampled and what does it imply?

Book
pp
44-66

Department of Electronics and Computer Science

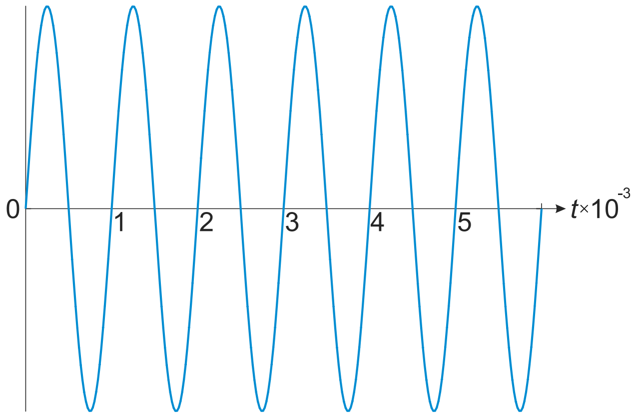
UNIVERSITY OF Southampton
School of Electronics and Computer Science

Content

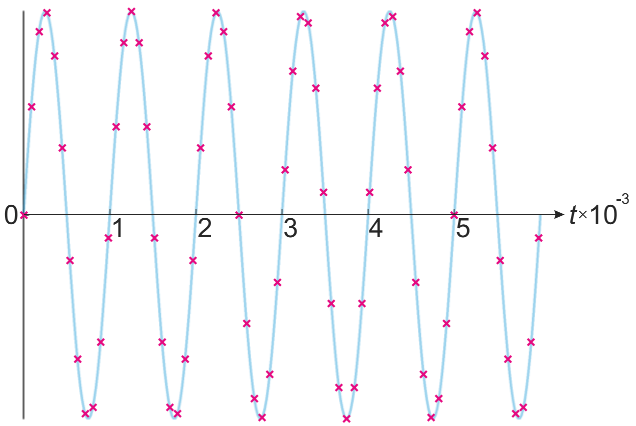
1. What can go wrong with sampling?
2. How does the discrete Fourier transform work, and help?

Sampling Signals

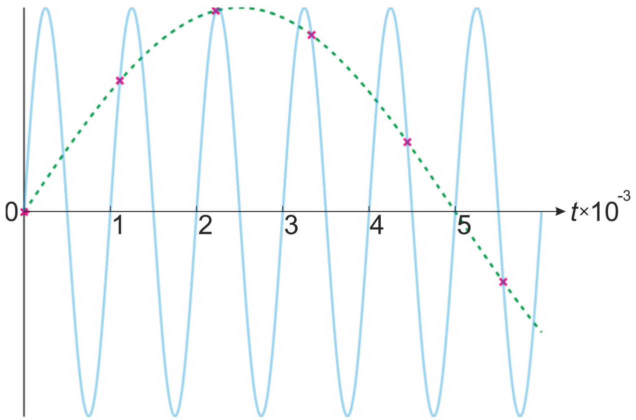
original continuous signal



good sampling



bad sampling (aliased)

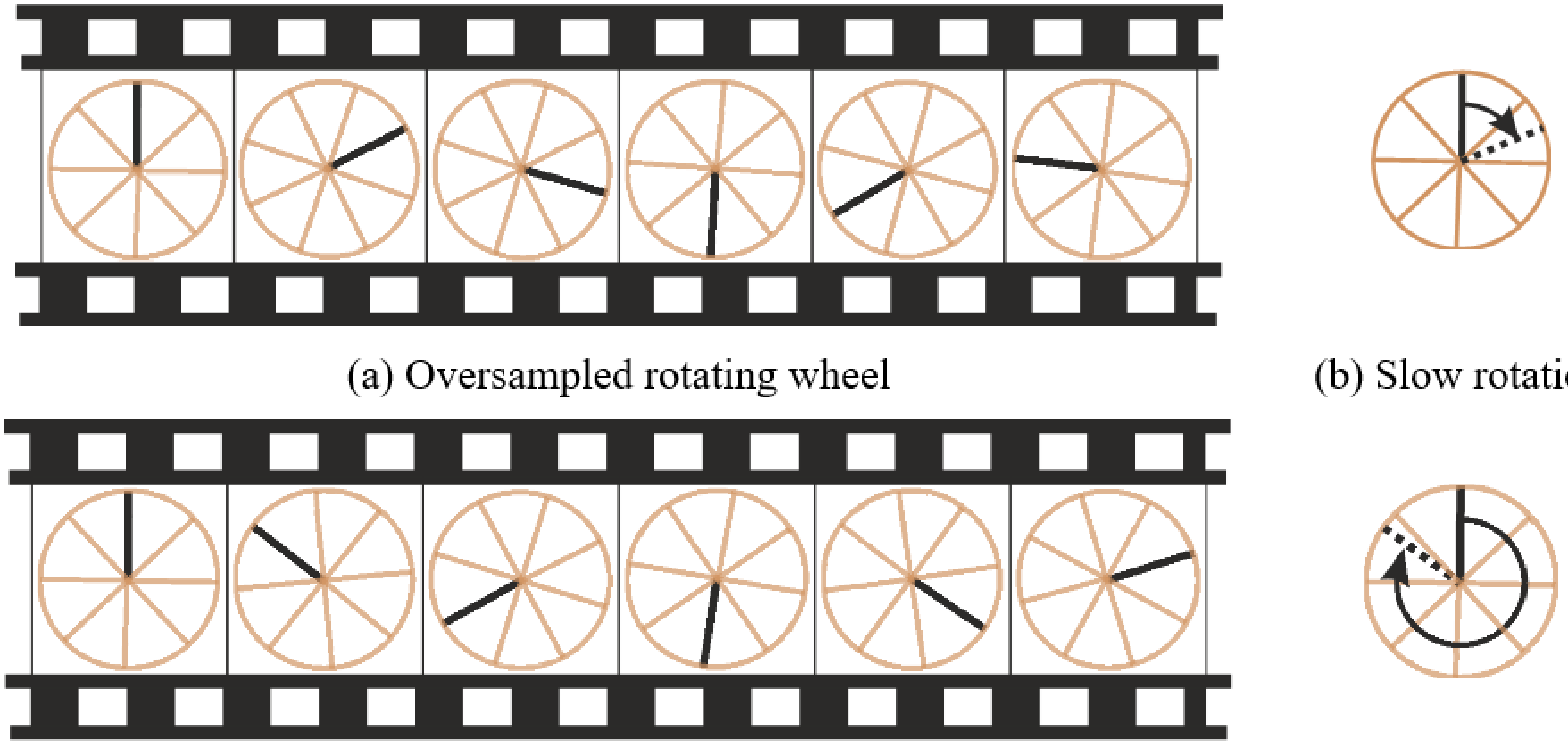


Aliasing in Sampled Imagery

(a) high resolution

(c) low resolution – aliased

Correct and Incorrect Apparent Wheel Motion



(a) Oversampled rotating wheel

(b) Slow rotation

(c) Undersampled rotating wheel

(d) Fast rotation

Figure 4.5 Correct and incorrect apparent wheel motion

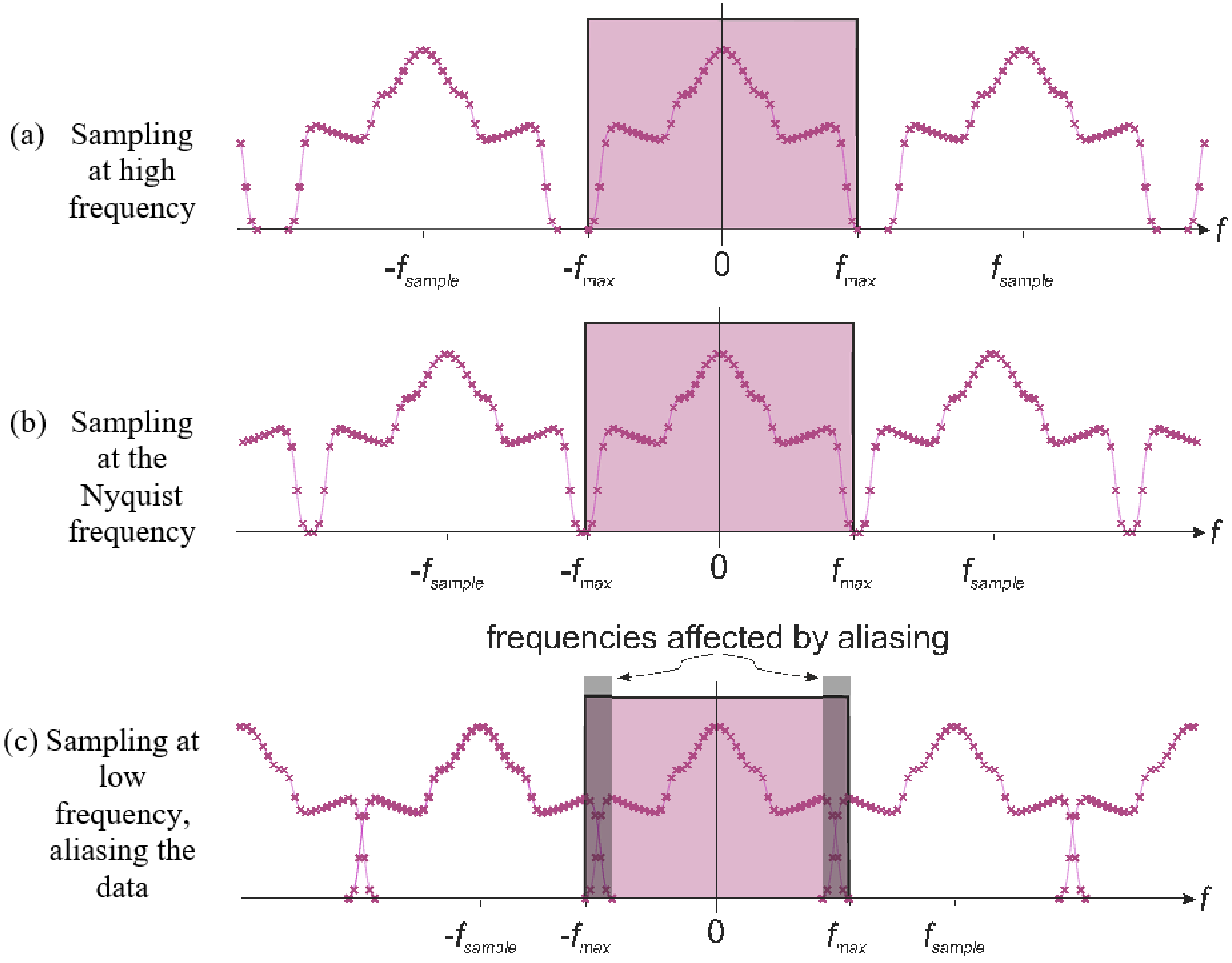
https://www.youtube.com/watch?v=e1EqXE06xr8

In the frequency domain

Spectra repeat

If sampling is just right, spectra just touch

Minimum sampling frequency = 2 × max



Sampling process in the frequency domain

Sampling theory

Nyquist's sampling theorem

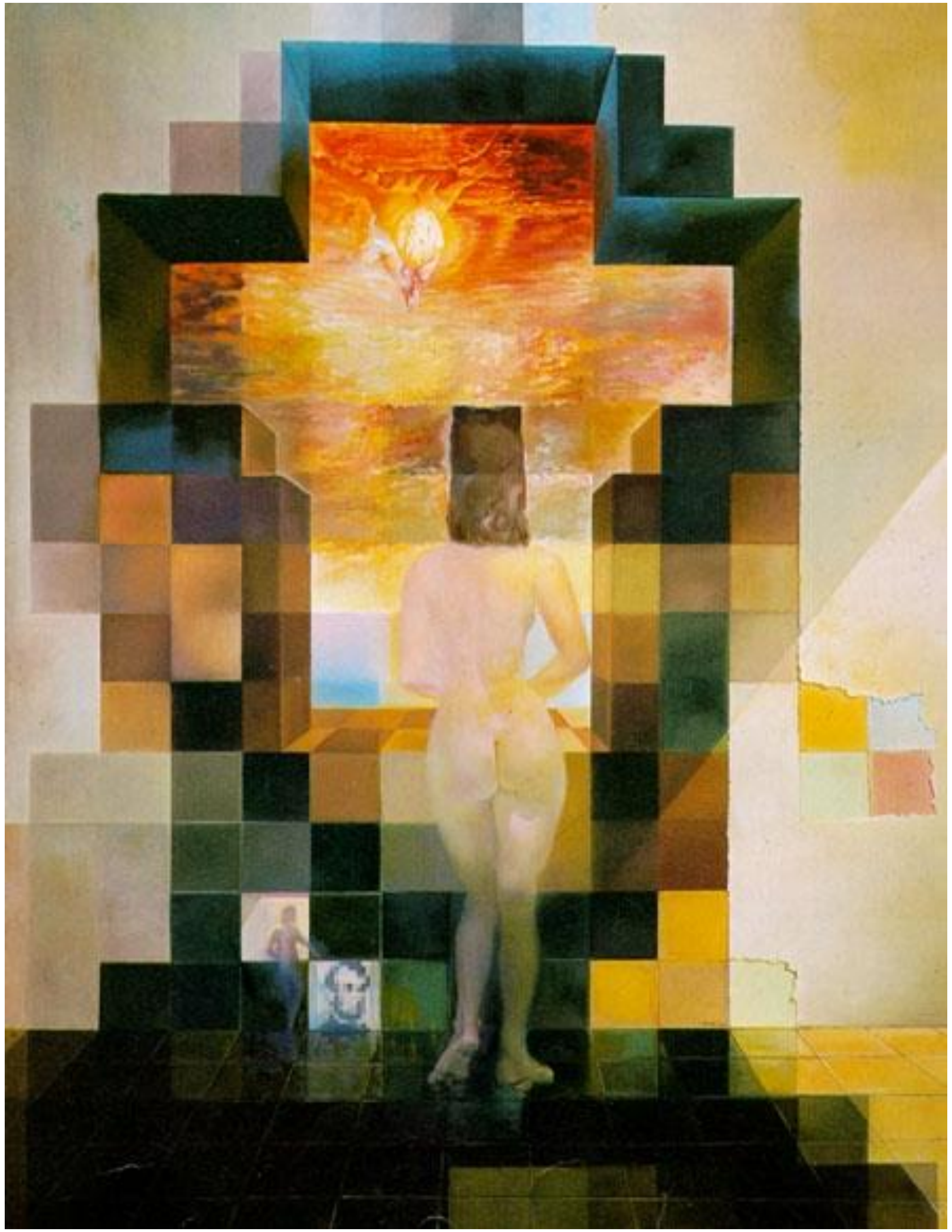
In order to be able to be able to reconstruct a signal from its samples we must sample at minimum at twice the maximum frequency in the original signal

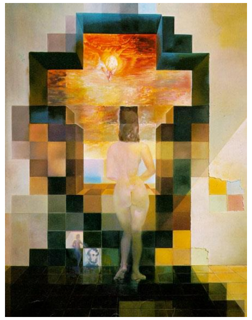
E.g. speech 6kHz, sample at 12 kHz

Video bandwidth (CCIR) is 5MHz

Sampling at 10MHz gave 576×576 images

Guideline: "two pixels for every pixel of interest"





https://www.pinterest.com/pin/275423333431517864/

1D Discrete Fourier transfrom

Discrete Fourier calculates frequency from data points

$$Fp_u = \qquad p_i$$

sampled frequency Fp_u

sampled points p_i

1D Discrete Fourier transfrom

Discrete Fourier calculates frequency from data points

$$Fp_u = \frac{1}{N}\sum_{i=0}^{N-1} p_i$$

sampled frequency Fp_u

sampled points p_i

N points

1D Discrete Fourier transfrom

Discrete Fourier calculates frequency from data points

$$Fp_u = \frac{1}{N}\sum_{i=0}^{N-1} p_i e^{-j\frac{2\pi}{N}iu}$$

sampled frequency Fp_u

sampled points p_i

N points

$$e^{-j\theta} = \cos\theta - j\sin\theta$$

1D Discrete Fourier transfrom

Discrete Fourier calculates frequency from data points

$$Fp_u = \frac{1}{N}\sum_{i=0}^{N-1} p_i e^{-j\frac{2\pi}{N}iu}$$

Comparison

$$Fp(\omega) = \int_{-\infty}^{\infty} p(t)e^{-j\omega t}\,dt$$

sampled frequency Fp_u

sampled points p_i

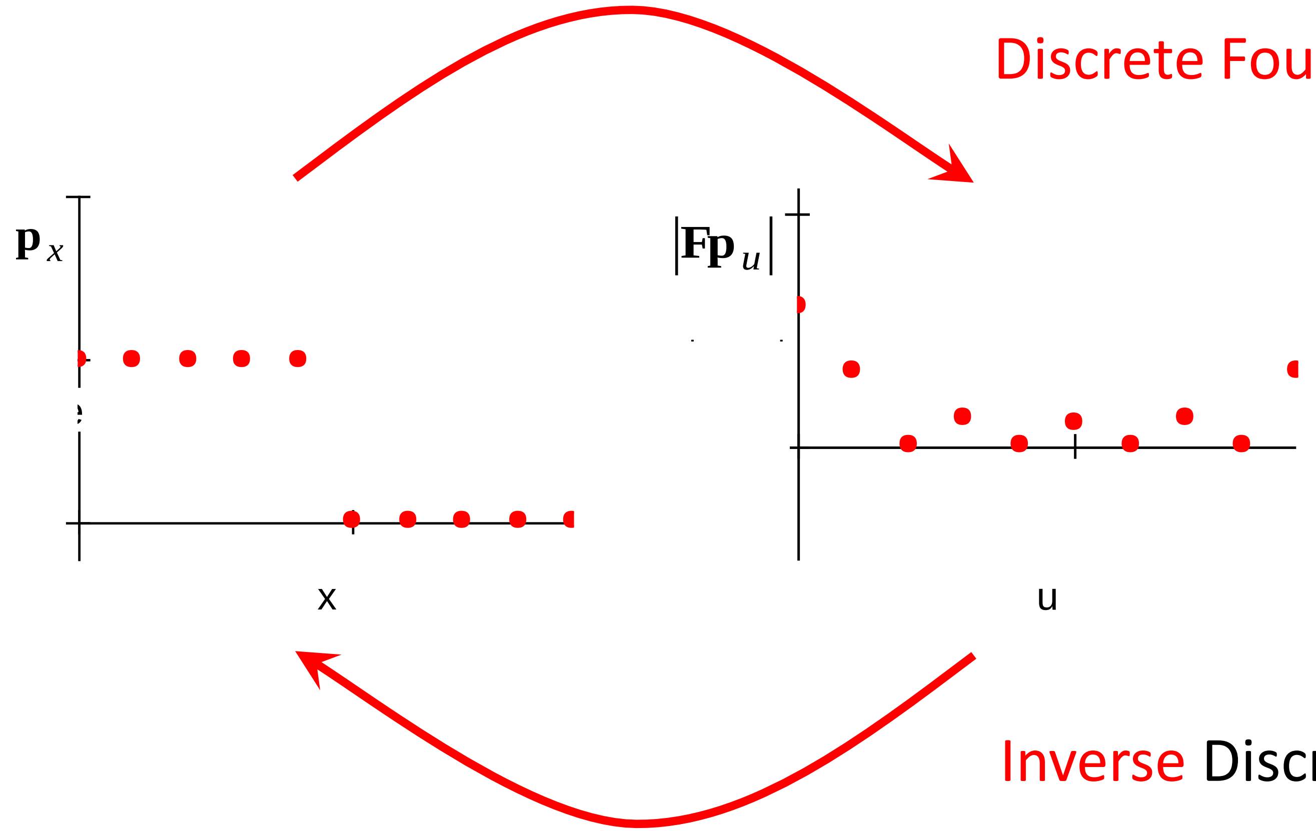
N points

$$e^{-j\theta} = \cos\theta - j\sin\theta$$

Fireside time

Why/ how did you (I) get into biometrics?

Transform Pair for Sampled Pulse

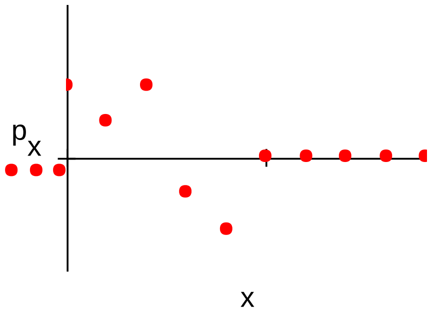


Discrete Fourier transform

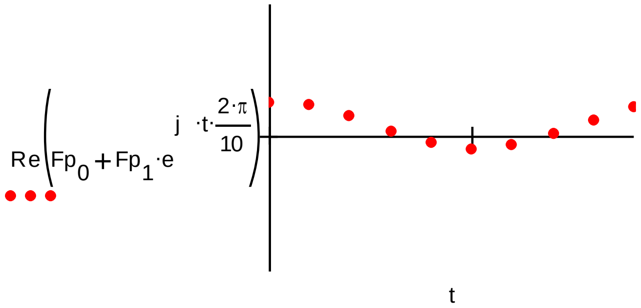
Inverse Discrete Fourier transform

(a) sampled pulse

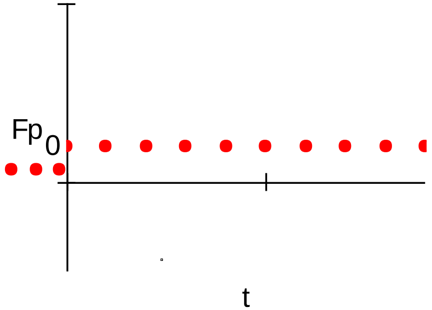
(b) DFT of sampled pulse



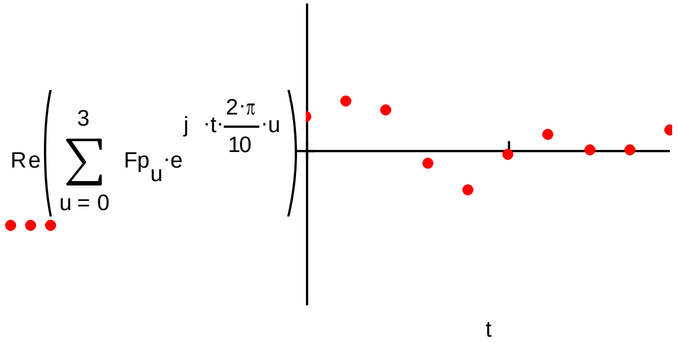
(a) original sampled signal



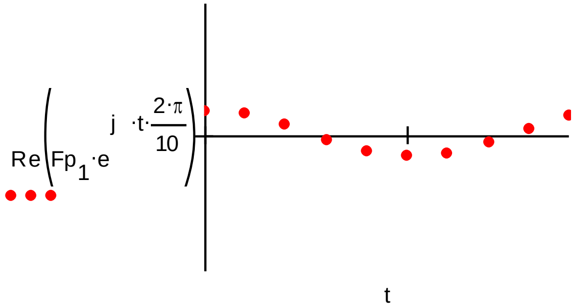
(b) first coefficient Fp_0

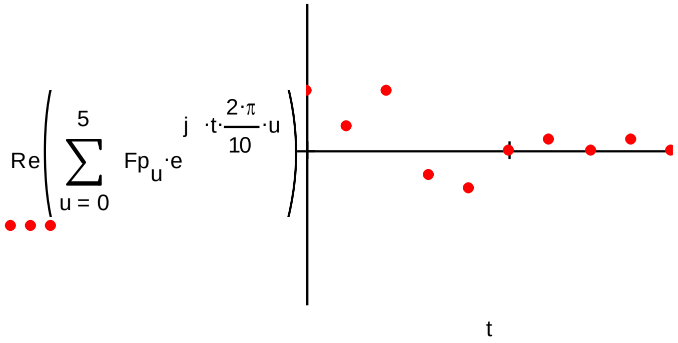


(c) second coefficient Fp_1



(d) adding Fp_1 and Fp_0





(e) adding Fp_0, Fp_1 , Fp_2 and Fp_3

(f) adding all six frequency components

signal reconstruction from its transform components

2D Fourier transform

Forward transform $$\mathbf{FP}_{u,v} = \frac{1}{N^2}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}\mathbf{P}_{x,y}\,e^{-j\left(\frac{2\pi}{N}\right)(ux+vy)}$$

where two dimensions of space, *x* and *y*

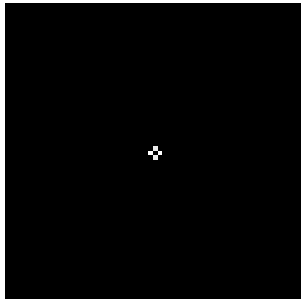
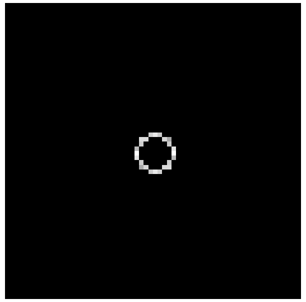
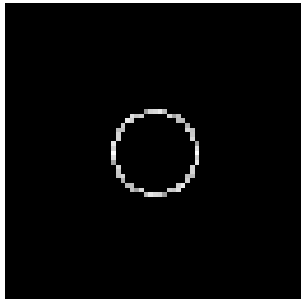
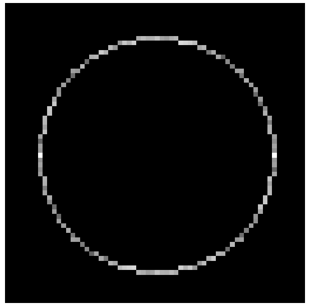
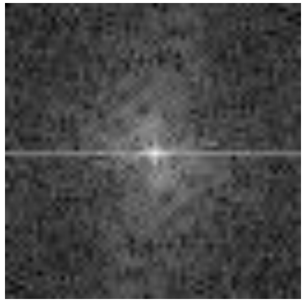
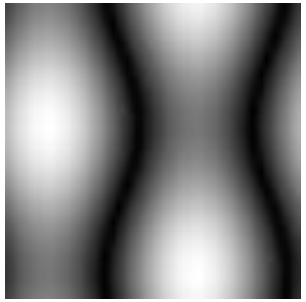
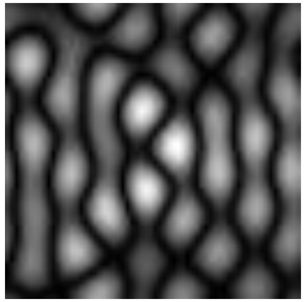
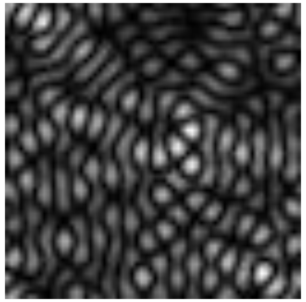
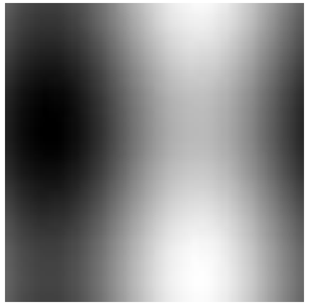
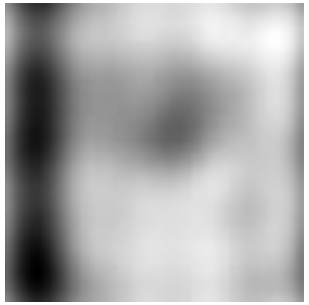
two dimensions of frequency, *u* and *v*

image *N*x*N* pixels $\mathbf{P}_{x,y}$

Inverse transform $$\mathbf{P}_{x,y} = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\mathbf{FP}_{u,v}\,e^{j\left(\frac{2\pi}{N}\right)(ux+vy)}$$

π??

Reconstruction

				
(a) transform radius 1 components	**(b)** transform radius 4 components	**(c)** transform radius 9 components	**(d)** transform radius 25 components	**(e)** complete transform
				
(f) image by radius 1 components	**(g)** image by radius 4 components	**(h)** image by radius 9 components	**(i)** image by radius 25 components	
				
(j) reconstruction up to 1st	**(k)** reconstruction up to 4th	**(l)** reconstruction up to 9th	**(m)** reconstruction up to 25th	**(n)** reconstruction with all

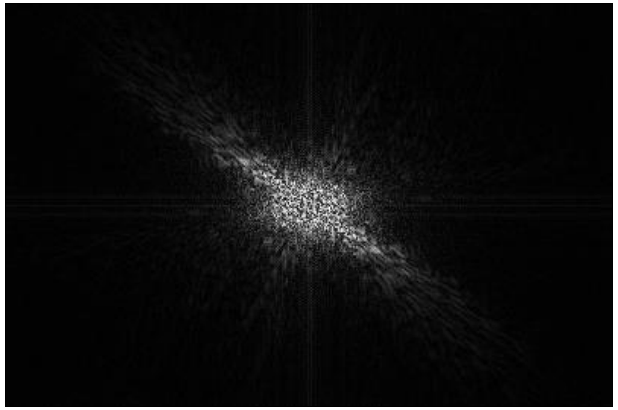
Implementation is via (Fast) FFT

```
while L<cols %iterate until log2(cols)-1 levels have been performed
  for j=1:2*L:cols %do all the points in L/2 batches
    for i=1:L %now do L butterflies
      upp(((j+1)/2)+i-1)= Fp(j+i-1)+Fp(j+L+i-1)*exp(-1j*2*pi*(i-1)/(L*2));
      low(((j+1)/2)+i-1)= Fp(j+i-1)-Fp(j+L+i-1)*exp(-1j*2*pi*(i-1)/(L*2));
    end
  end
  for j=1:2*L:cols %copy the components across, to the right places
    for i=1:L
      Fp(j+i-1)=upp(((j+1)/2)+i-1);
      Fp(j+L+i-1)=low(((j+1)/2)+i-1);
    end
  end
L=L*2; %and go and do the next level (up)
end
```

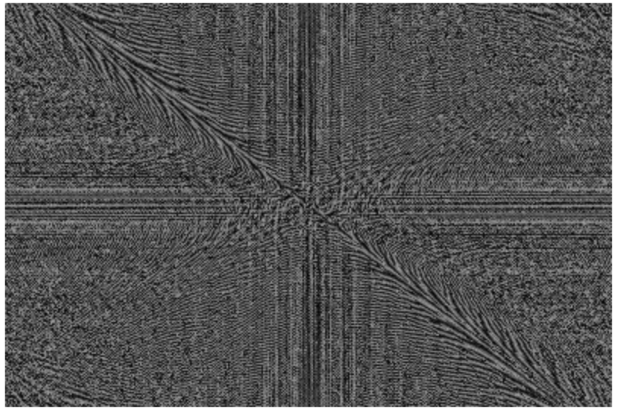
(this is a 1-D FFT)

Shift invariance

(a) original image

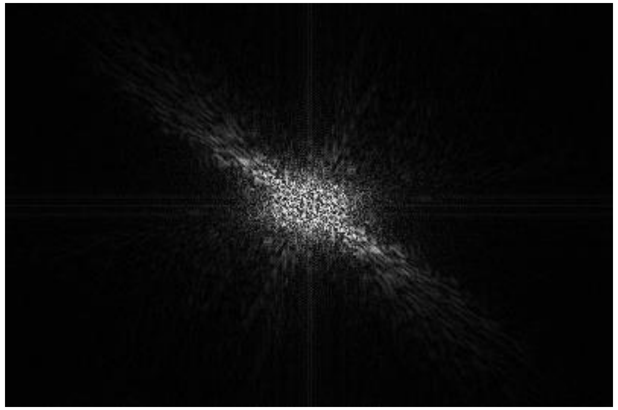


(b) magnitude of Fourier transform of original image

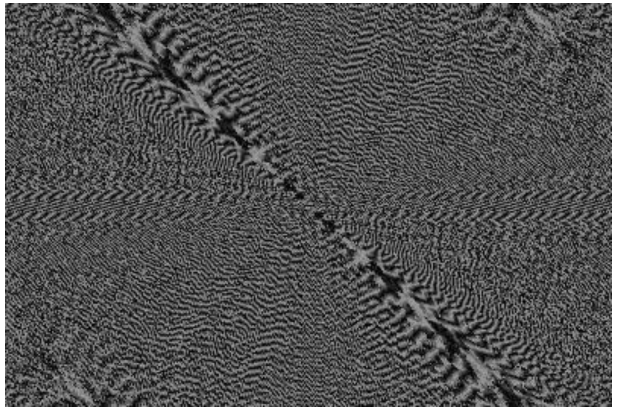


(c) phase of Fourier transform of original image

(d) shifted image

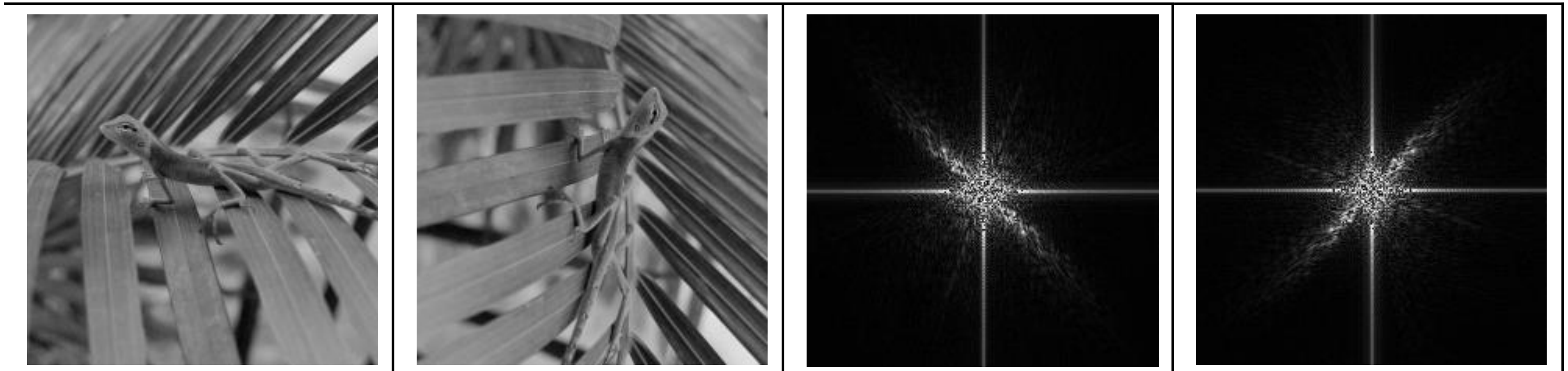


(e) magnitude of Fourier transform of shifted image



(f) phase of Fourier transform of shifted image

Rotation

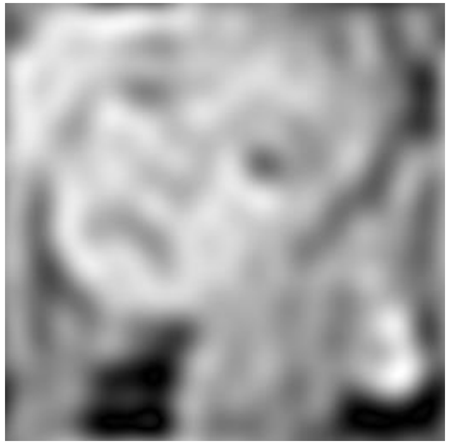


(a) original image	**(b)** rotated image	**(c)** transform of original image	**(d)** transform of rotated image

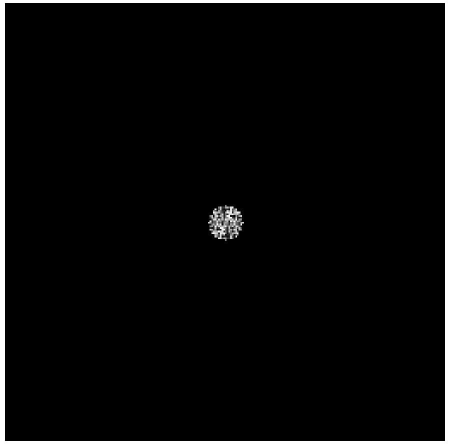
$$\mathbf{FP}_{u,v} = \frac{1}{N}\sum_{y=0}^{N-1}\sum_{x=0}^{N-1}\mathbf{P}_{x,y}\,e^{-j\left(\frac{2\pi}{N}\right)(uy+vx)}$$

Filtering

Fourier gives access to
frequency components

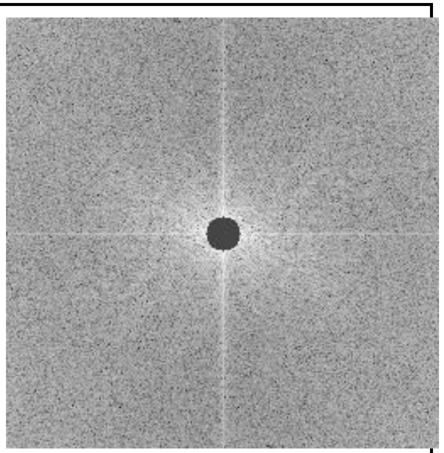


(a) low-pass filtered image



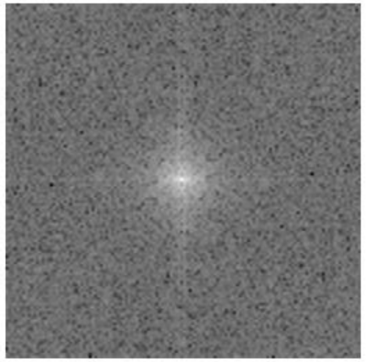
(b) low-pass filtered transform

(c) high-pass filtered image

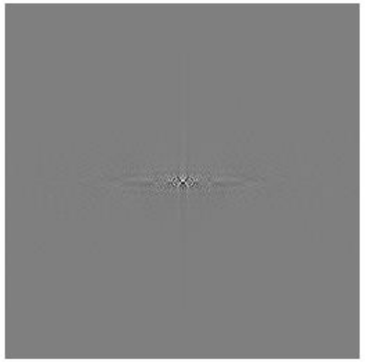


(d) high-pass filtered transform

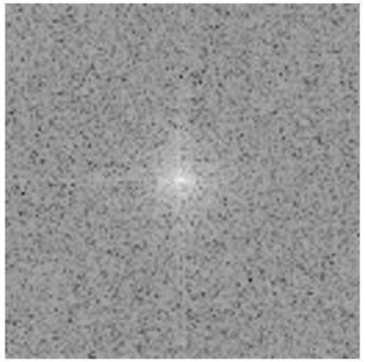
Other transforms



(a) Fourier transform magnitude



(b) discrete cosine transform

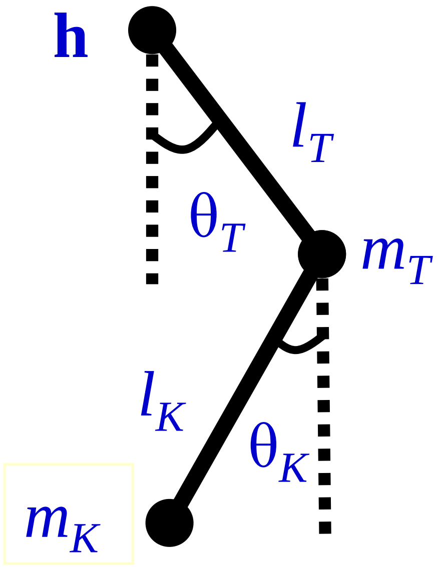


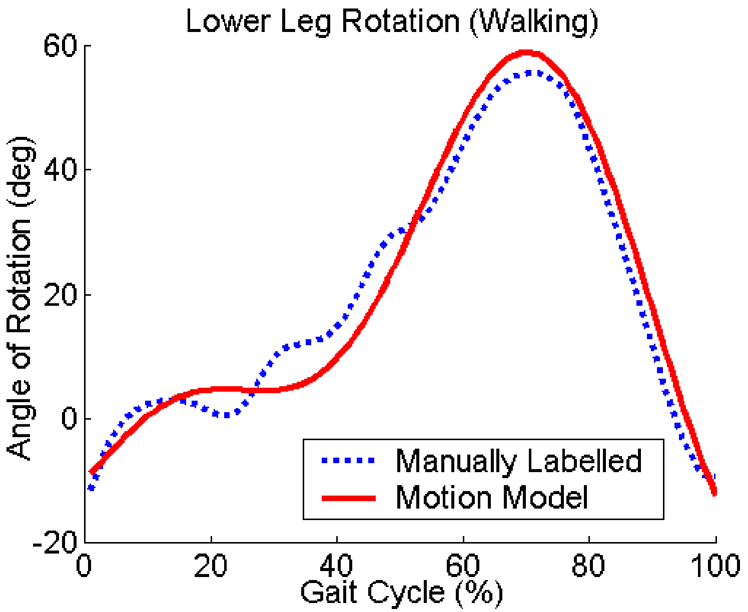
(c) Hartley transform

Comparing Transforms

Modelling Gait(s)

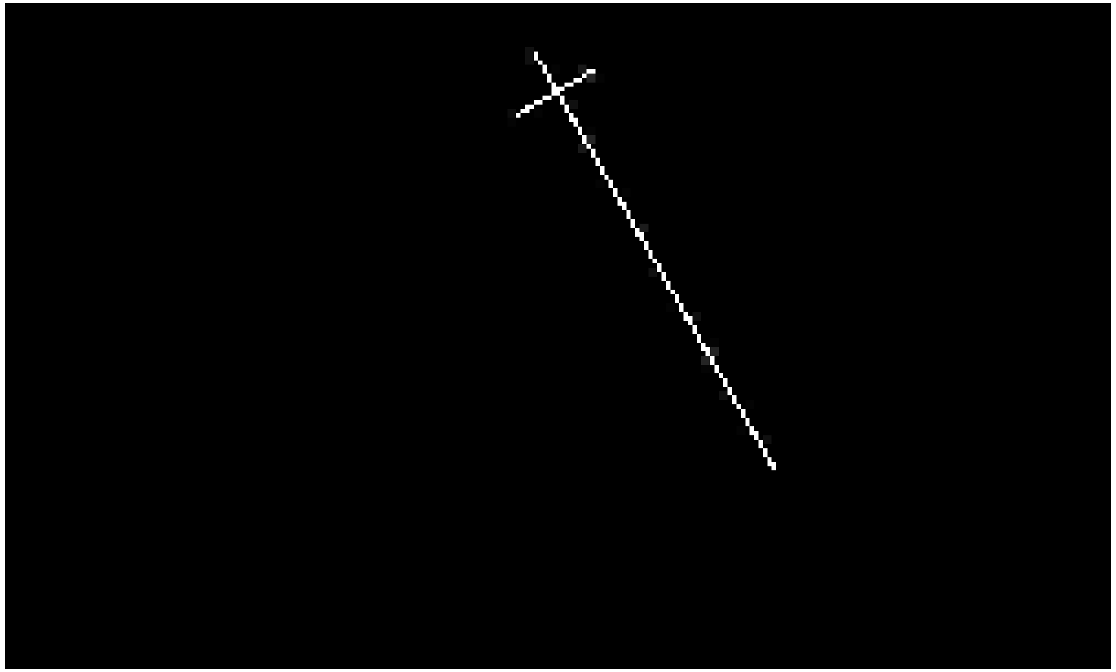
- Extended pendular thigh-model, based on angles
- Uses forced oscillator/ bilateral symmetry/ phase coupling





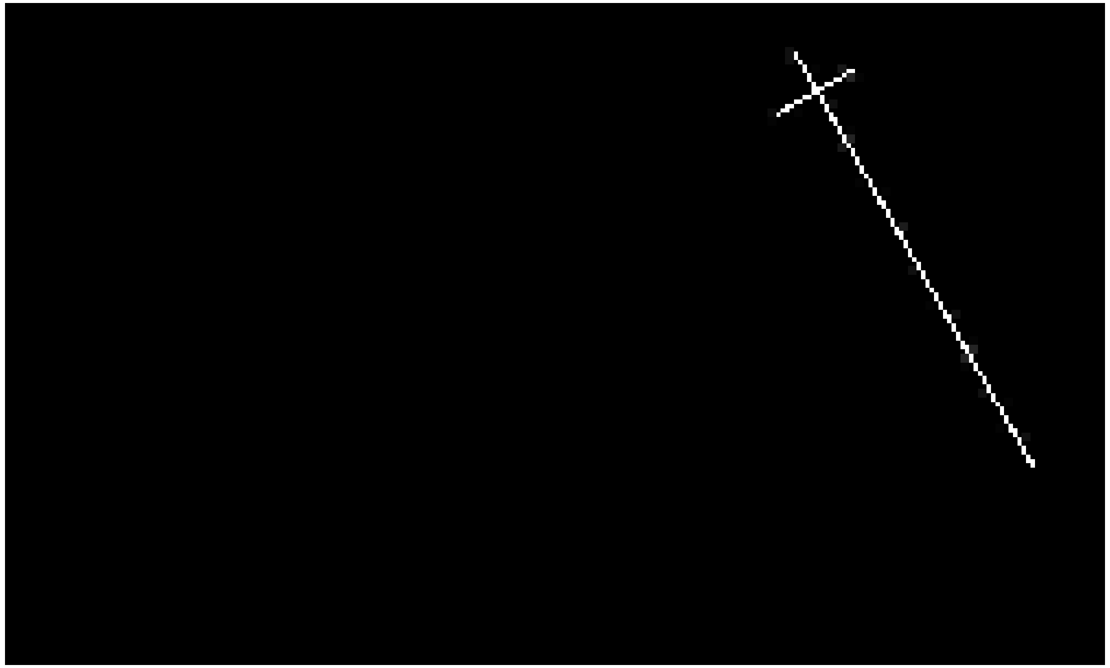
Yam, Nixon and Carter, *AVBPA* 2001, *Patt. Recog.* 2004

Modeling the Thigh's Motion 1



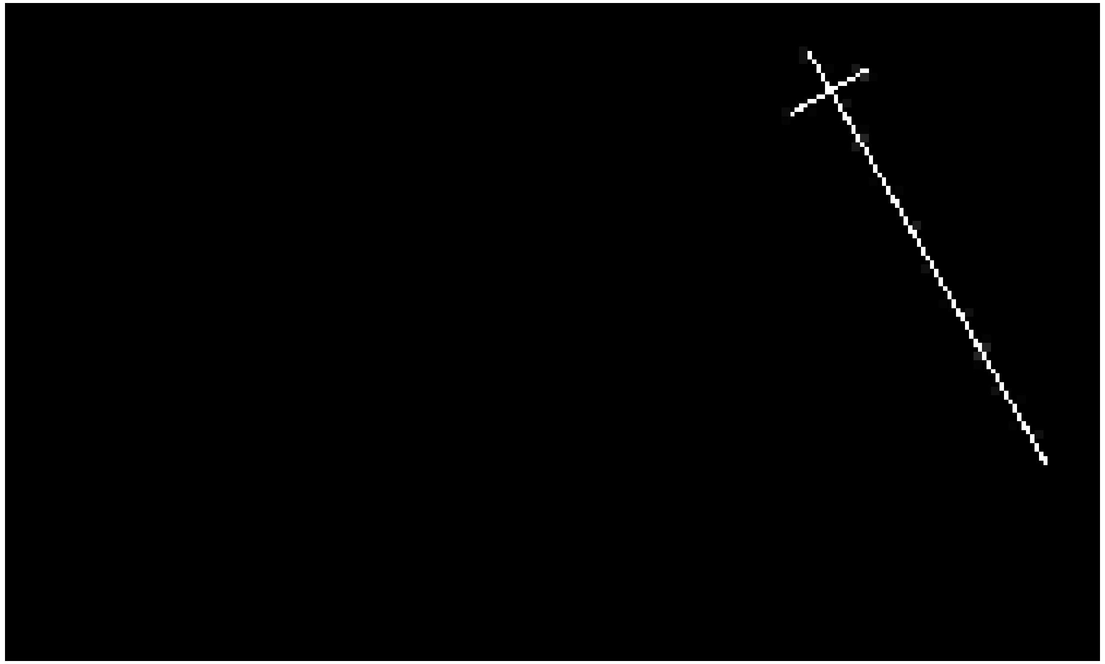
$$vs_x(t) = A\cos(\omega t + \phi)$$

Modeling the Thigh's Motion 2



$$vh_x(t) = Vx + A\cos(\omega t + \phi)$$

Modeling the Thigh's Motion 3



$$\phi(t) = a_0 + \sum_{k=1}^{N} \left[b_k \cos\left(k\omega_0 t + \psi \right) \right]$$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Validity?

Applications of 2D FT

- Understanding and analysis
- Speeding up algorithms
- Representation (invariance)
- Coding
- Recognition/ understanding (e.g. texture)

Takeaway time

1 – need to sample at a high enough frequency

2 – aliasing corrupts image information

3 – discrete Fourier allows analysis and understanding

4 – Fourier has many properties and advantages

…. but it's complex. So we'll move on to processing images