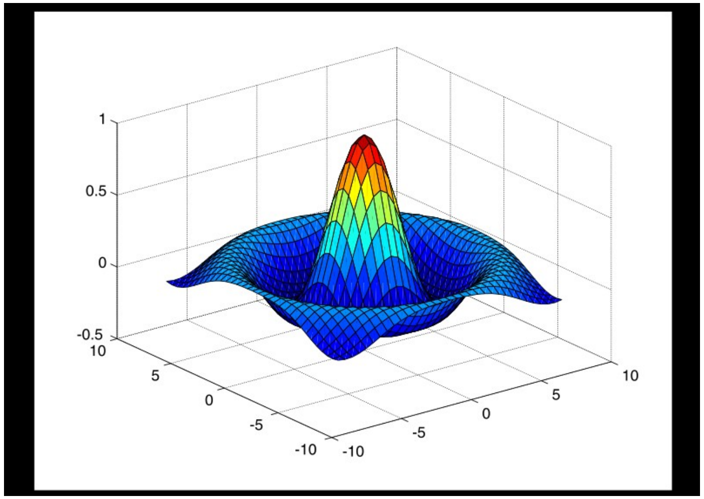
INF555

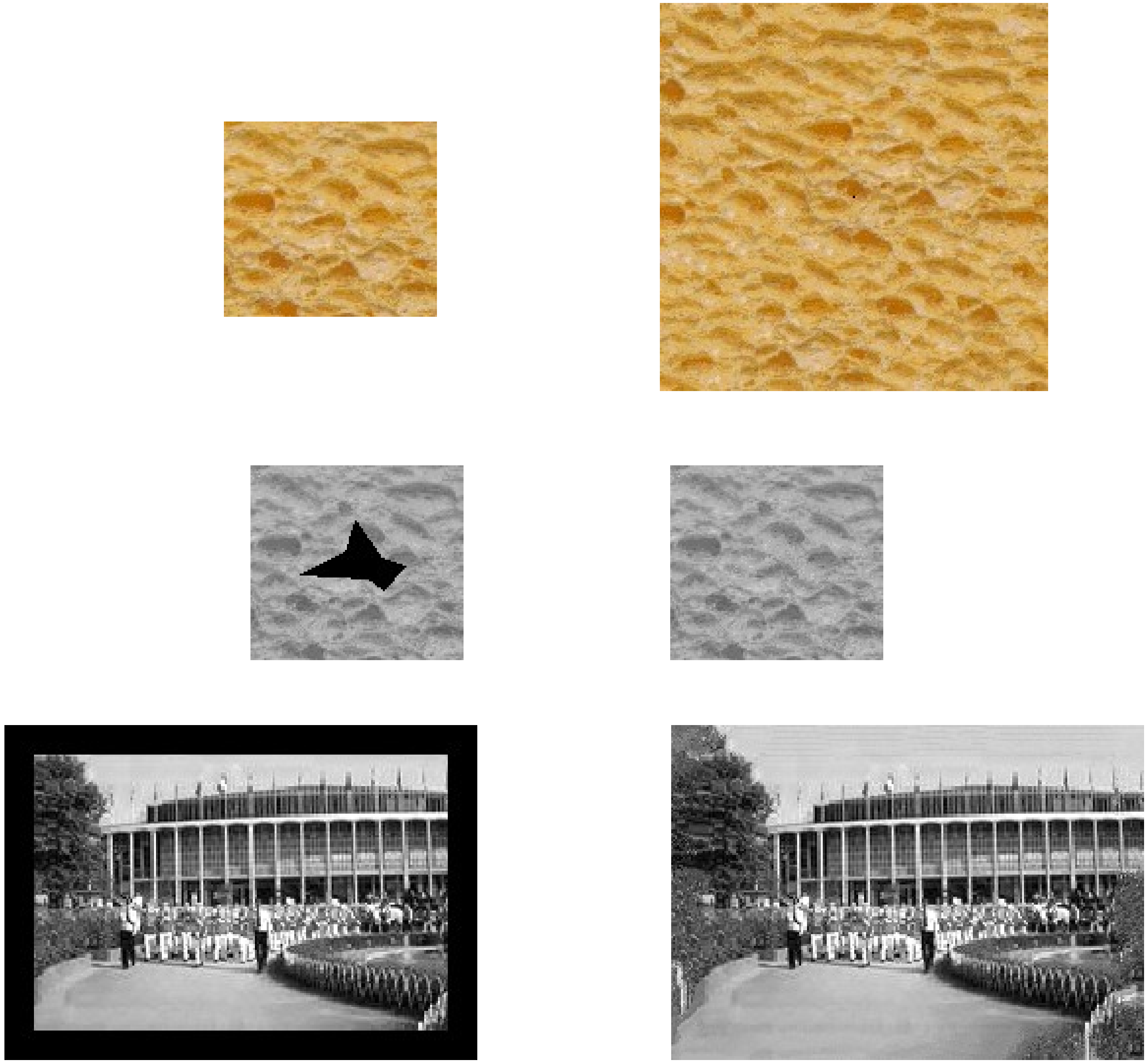


Fundamentals of 3D

Lecture 6:

Metric ball trees/Texture synthesis
Advanced coordinate pipelines
Fourier analysis/interpolation

Frank Nielsen
nielsen@lix.polytechnique.fr



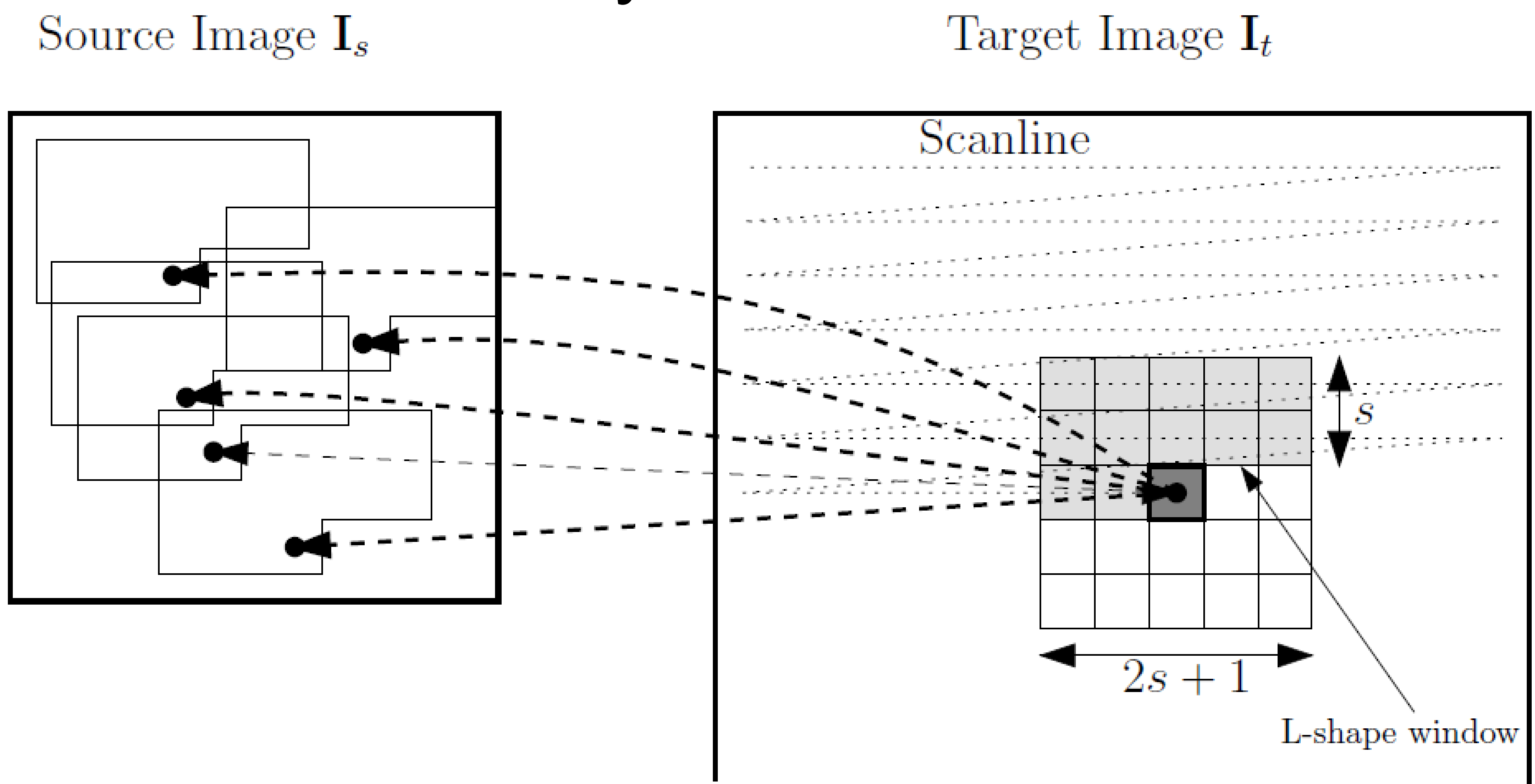
http://graphics.cs.cmu.edu/people/efros/research/EfrosLeung.html

``Texture Synthesis by Non-parametric Sampling''

Alexei A. Efros and Thomas K. Leung

IEEE International Conference on Computer Vision (ICCV'99),

Stochastic texture synthesis

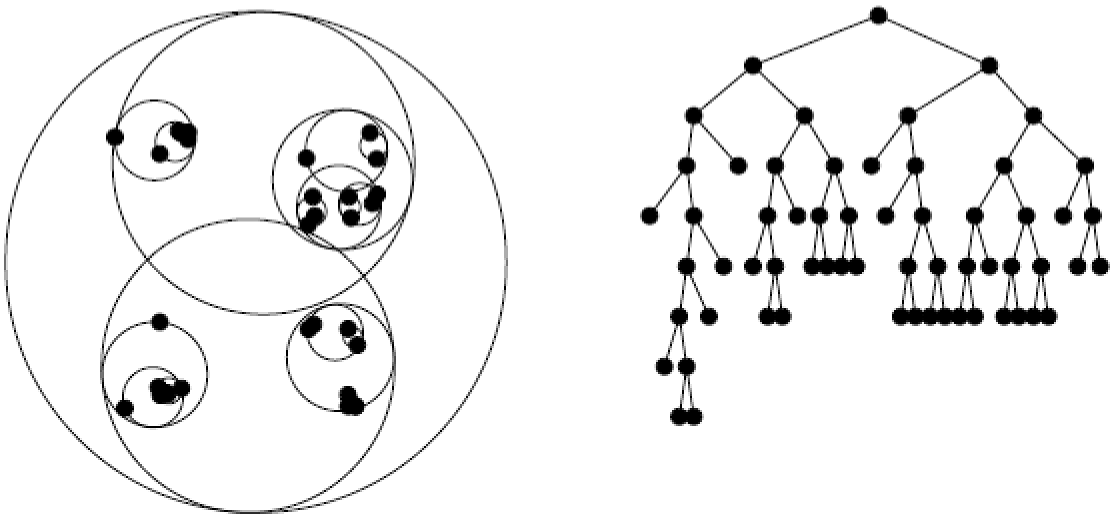


$$\mathrm{SSD}(x_s, y_s; x_t, y_t) = \sum_{l=-s}^{s} \sum_{c=-s}^{s} \mathrm{LShape}(l, c) \left(\mathbf{I}_s[x_s + c, y_s + l] - \mathbf{I}_t[x_t + c, y_t + l]\right)^2$$

$$(x_s, y_s) = \mathrm{argmin}_{(x,y)\in \mathbf{I}_s} \mathrm{SSD}(x, y; x_t, y_t).$$

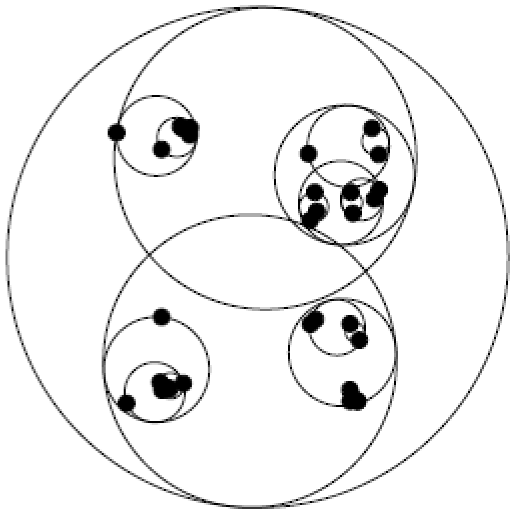
Fast nearest neighbor queries in high dimensions

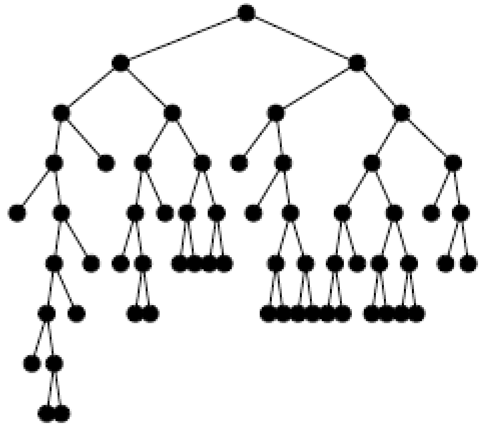
Ball tree data structures for nearest neighbor search



Compute a k-means on S with k=2
Split S into S1 and S2 according to the two centroids
Perform recursion on S1, and S2 until |S1|<n0 and |S2|<n0

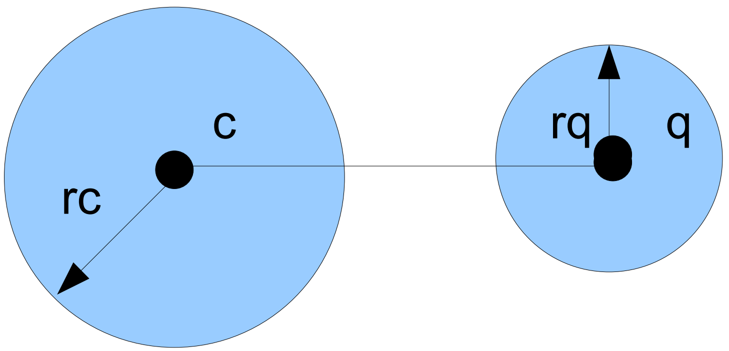
Nearest neighbor queries using ball trees





Pruning some of the nodes:
Let NN(q) denote the current best nearest neighbor of q

if ||q-c||-rq> rc then PRUNE (do not explore the subtree)



At leaves, perform the naive linear search, and potentially update NN(q)

Careful seeding for k-means: Perform just a careful initialization!!!

Interpolate between the two methods:

Let $D(x)$ be the distance between x and the nearest cluster center. Sample proportionally to $(D(x))^\alpha = D^\alpha(x)$

Original Lloyd's: $\alpha = 0$

Furthest Point: $\alpha = \infty$

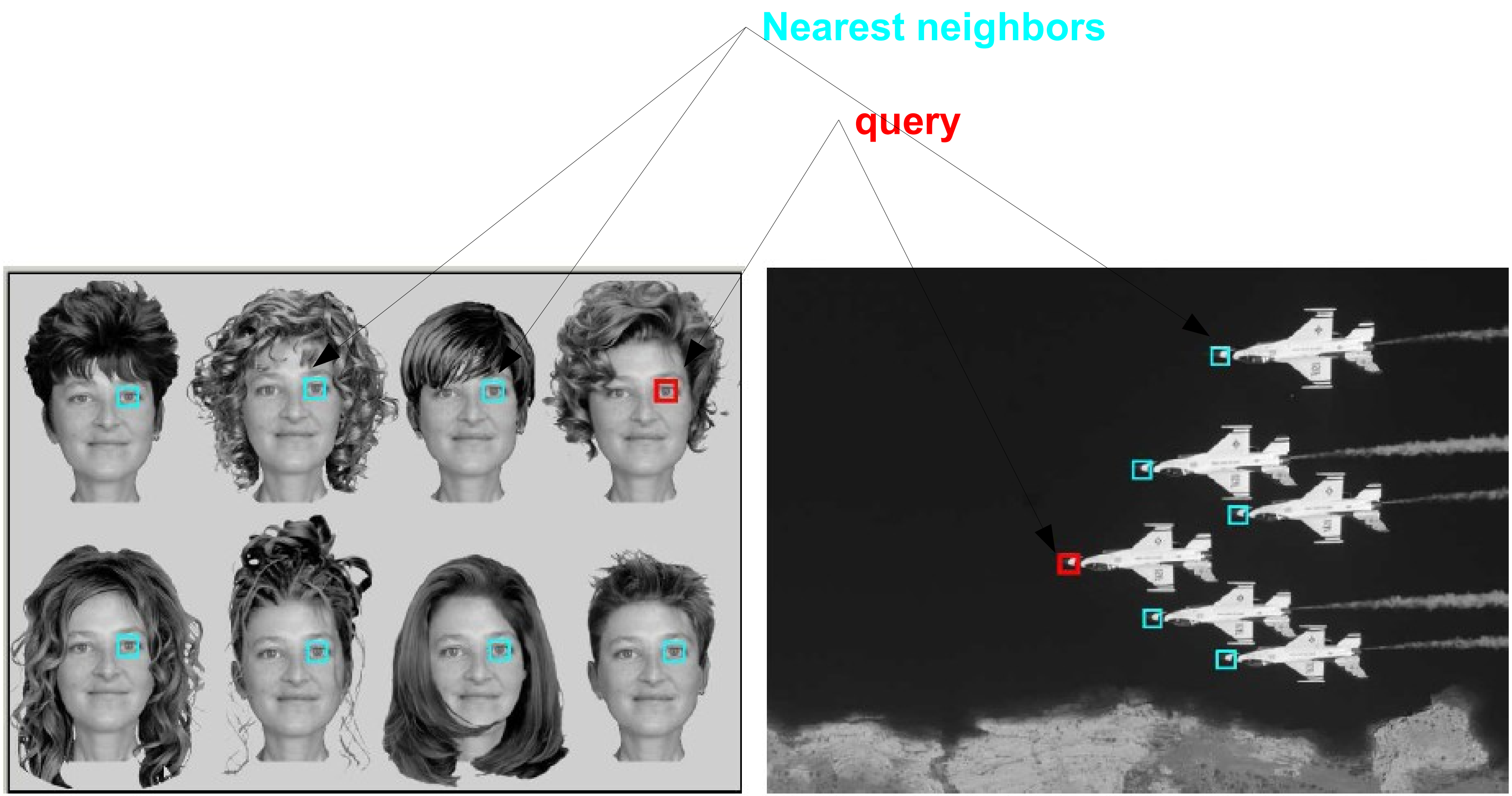
k-means++: $\alpha = 2$

1a. Choose an initial center c_1 uniformly at random from $\mathcal{X}$.

1b. Choose the next center c_i, selecting $c_i = x' \in \mathcal{X}$ with probability $\frac{D(x')^2}{\sum_{x \in \mathcal{X}} D(x)^2}$.

1c. Repeat Step 1b until we have chosen a total of k centers.

Theorem: k-means++ is $\Theta(\log k)$ approximate in expectation

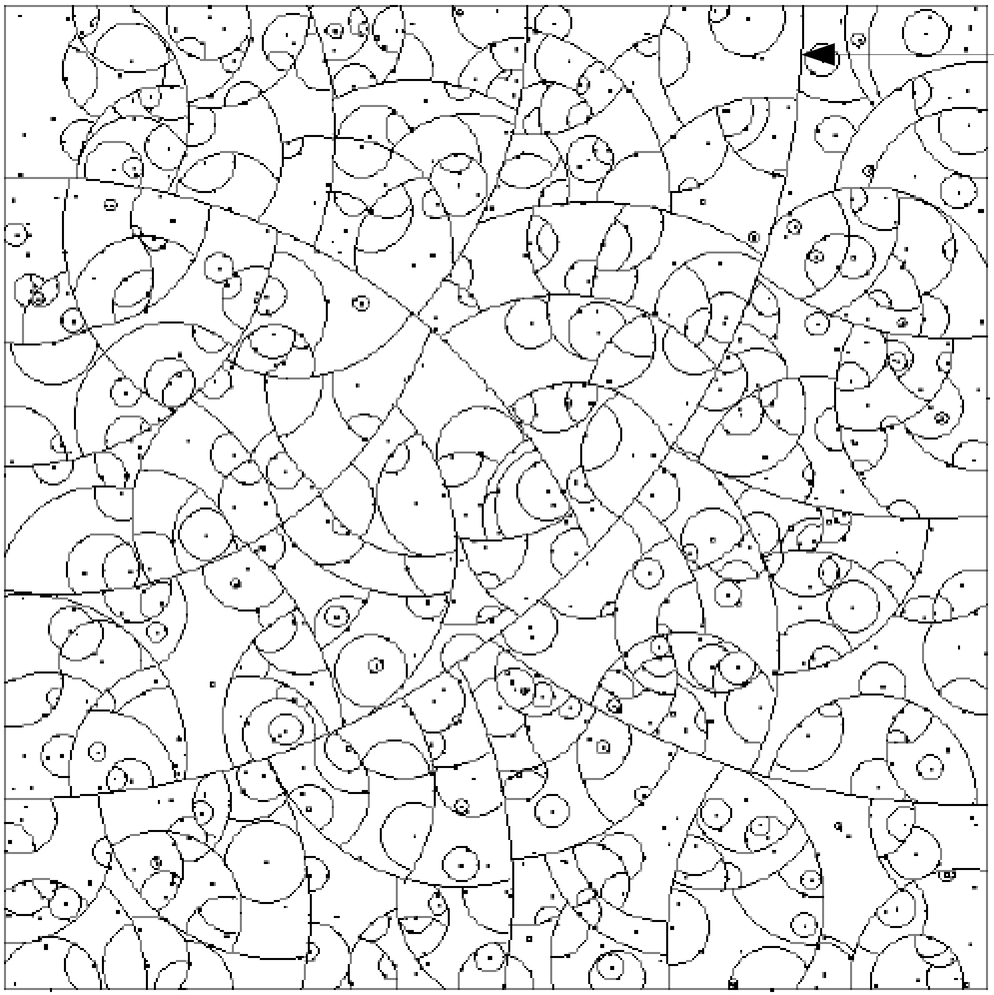


21x21 patches.

Vp-trees worked best (=fastest) for image patches.... [ECCV'08]

Vantage point trees (or vp-trees)

Partition the data according to a vantage point and a distance threshold
Relative distances are thus used.



First split

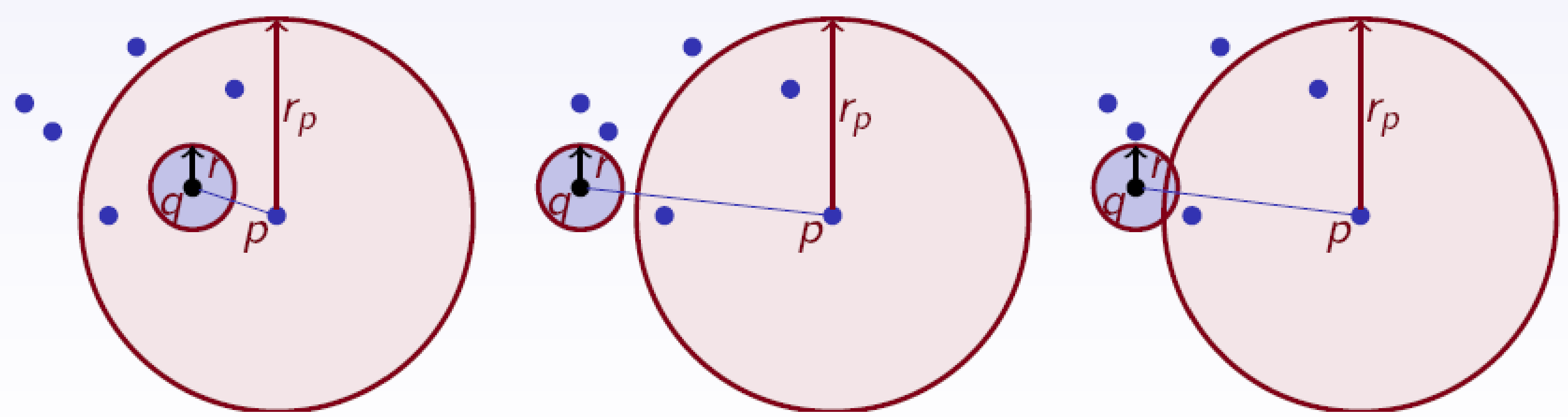
Split from a vantage point:
Inner part
Outer part

do split recursion

Vantage point trees: Pruning condition

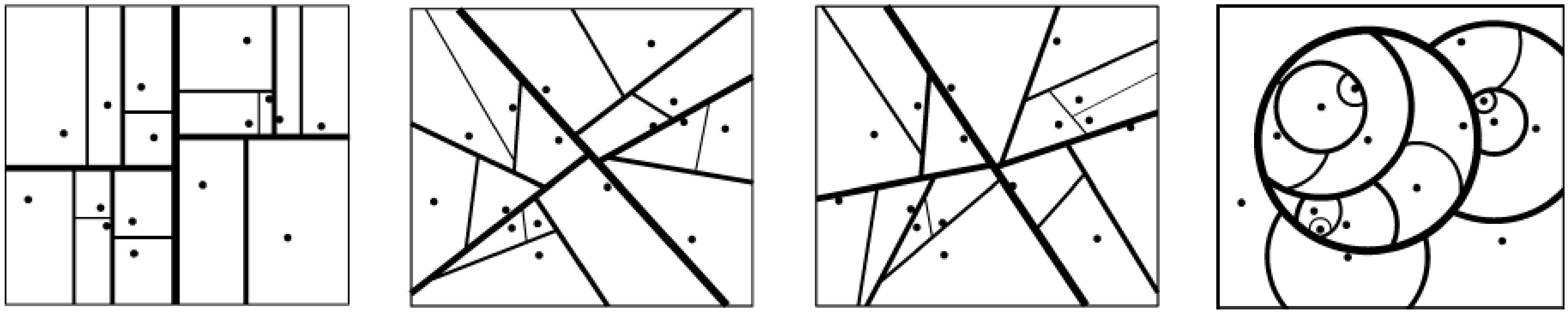
If $d(q,p) > r_p + r$ prune the inner branch
If $d(q,p) < r_p - r$ prune the outer branch

For $r_p - r \leq d(q,p) \leq r_p + r$ we have to inspect both branches

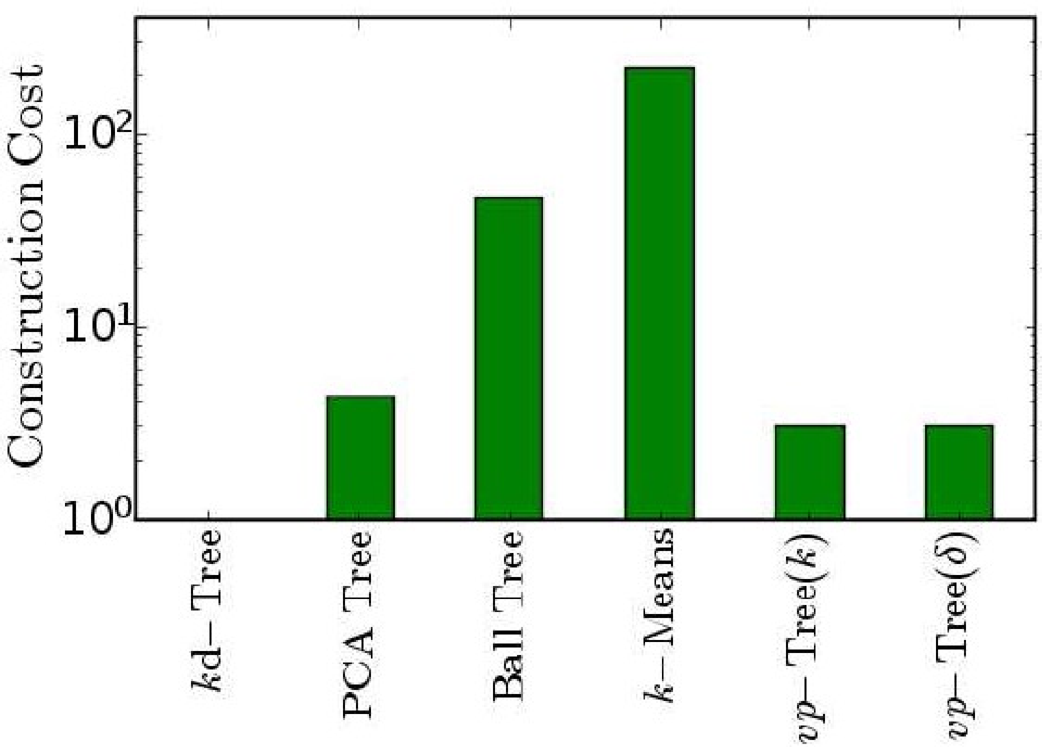


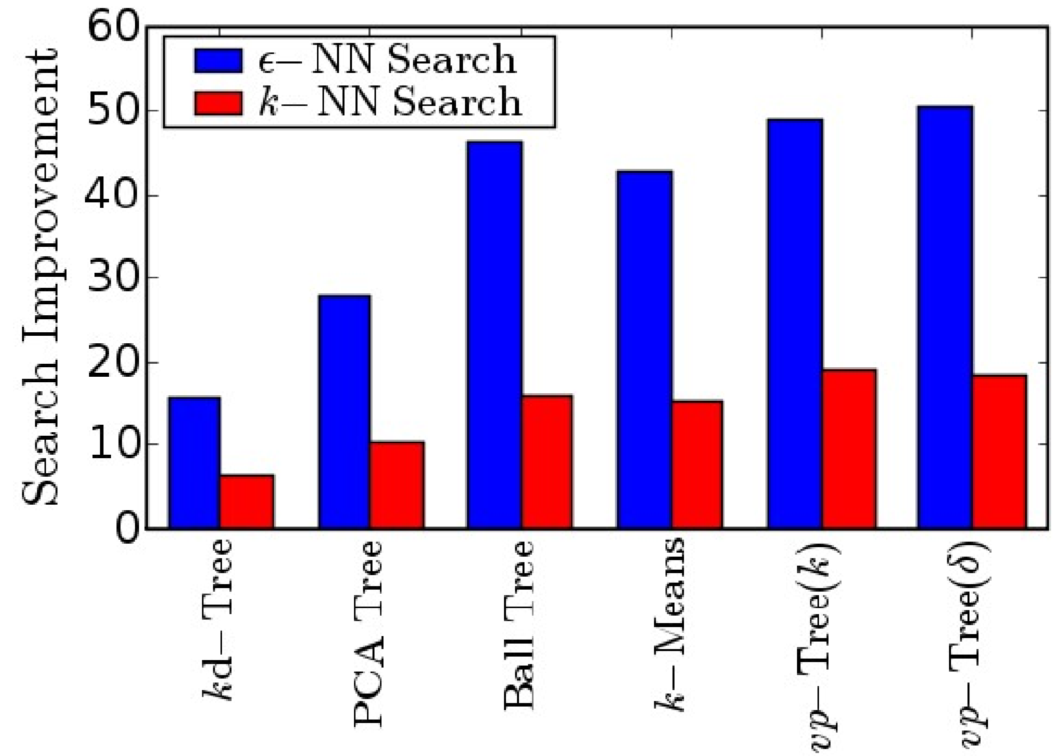
Prune outer **Prune inner** **Cannot prune**

Many ways to partition the point sets



(a) kd-Tree (b) PCA Tree (c) Ball Tree (d) vp-Tree

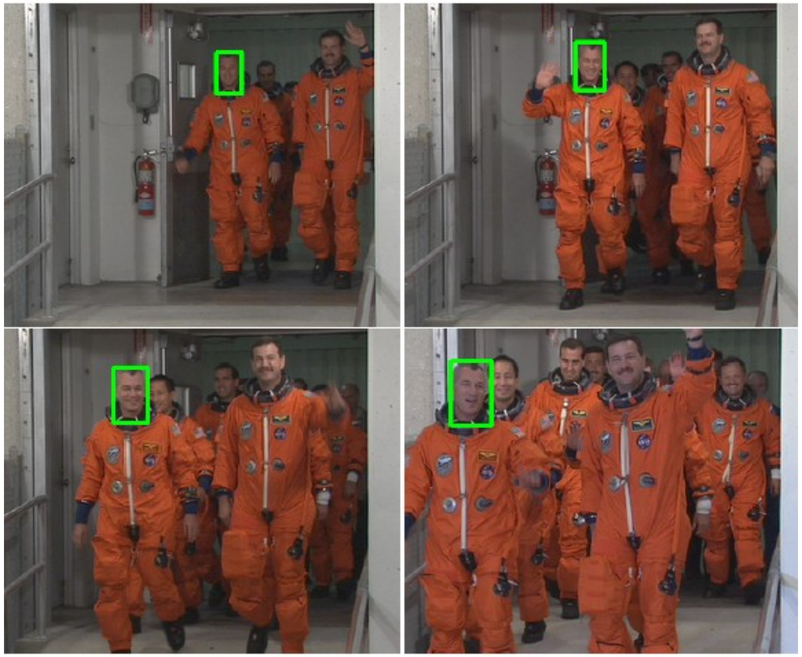


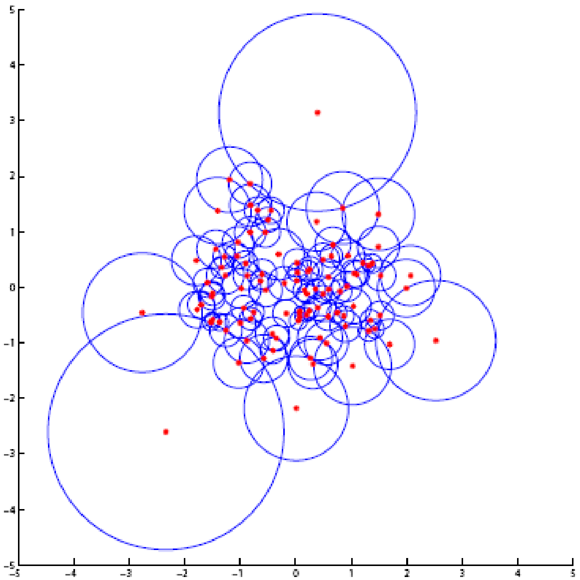


GPGPU: General Purpose GPU

GPU cores

Tracking
ROIs
(region of interest)





Descripteur	Dimension	ANN-C++	BF-CUDA	Gain
C	3	1m 33s	53s	1.8
CP	5	2m 05s	1m 05s	1.9
CG	5	2m 35s	1m 07s	2.3
CGP	7	4m 27s	1m 19s	3.3
C_3	11	6m 40s	1m 17s	5.2
C_3P	13	5m 43s	1m 12s	4.8

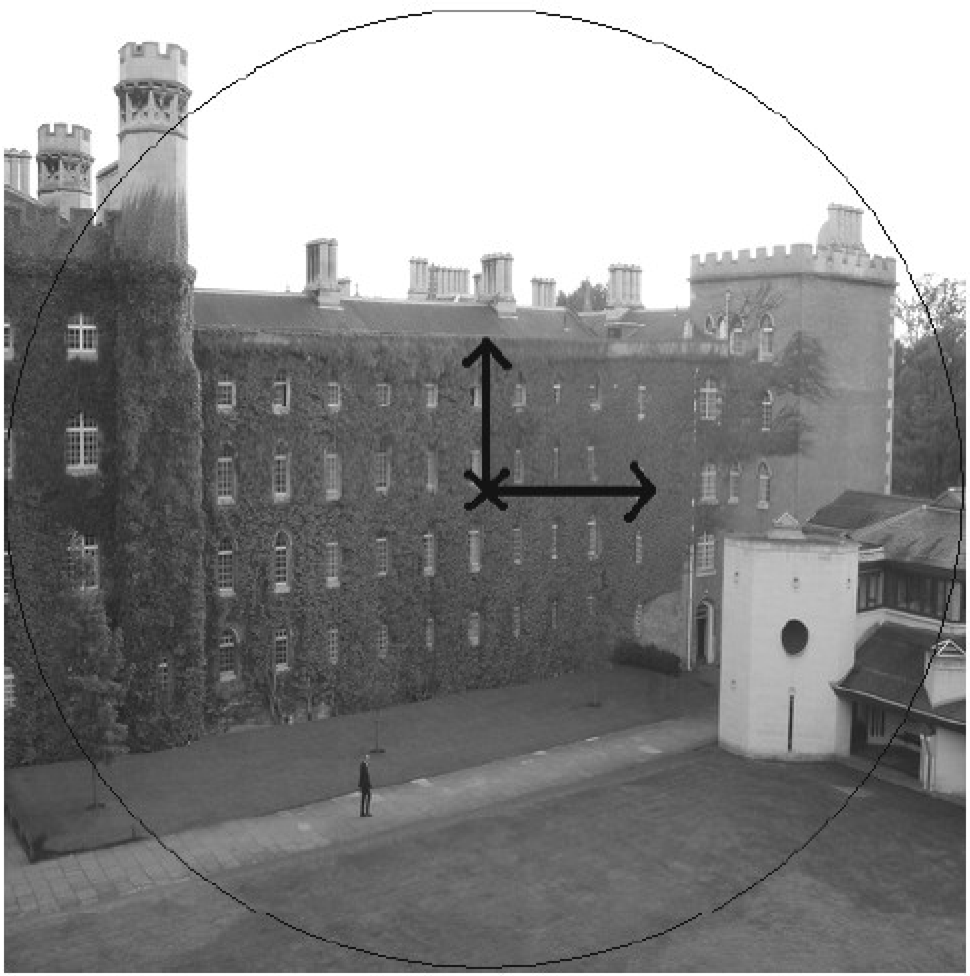
Distance au k=2 plus proche voisin

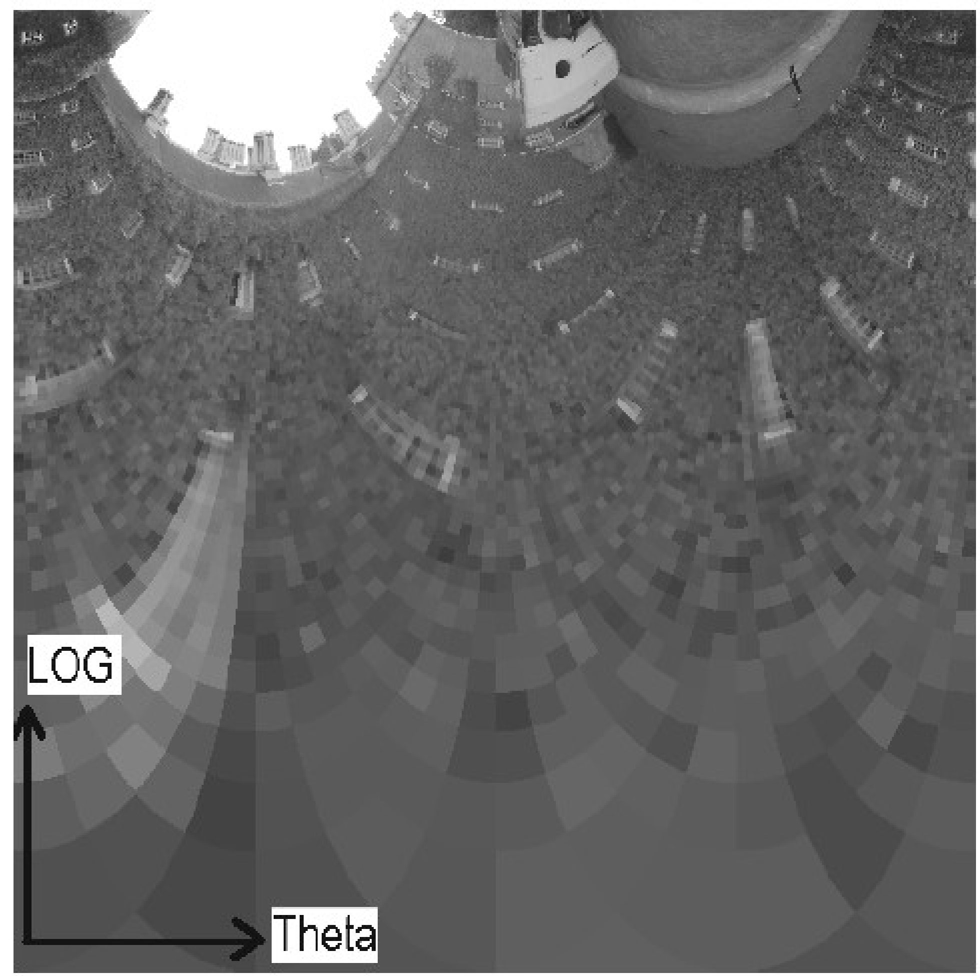
C: couleur (YUV)
C3: 3x3 neigh in Y
G: gradient
P: position
k=3

Log-Polar coordinates

$$\rho = \log\sqrt{x^2 + y^2},$$
$$\theta = \arctan\frac{y}{x}.$$

$$\rho = \log\sqrt{(x - x_o)^2 + (y - y_o)^2},$$
$$\theta = \arctan\frac{y - y_o}{x - x_o}.$$



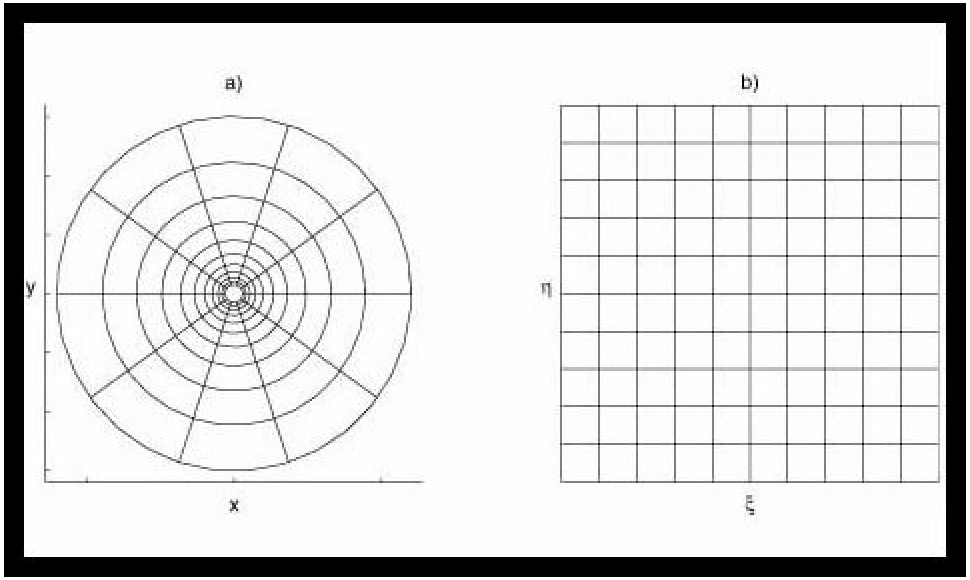


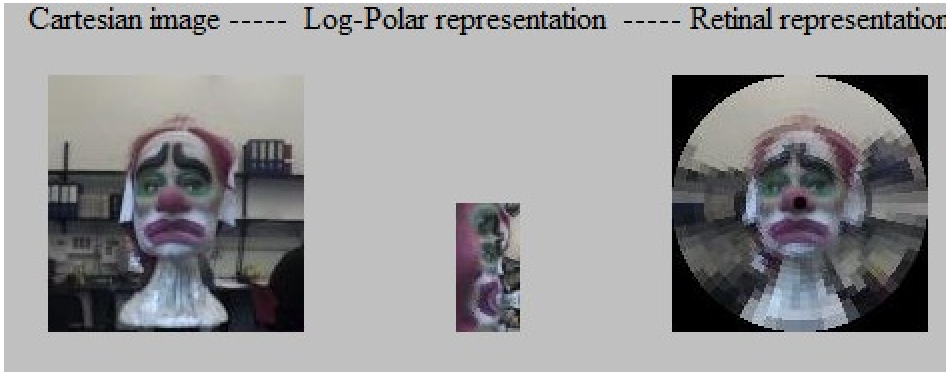
Log-Polar coordinates

Scales and rotations become translations

$$\mathbf{S}_s\mathbf{x} = (sx, sy) \longrightarrow (\log s + \rho(\mathbf{x}), \theta(\mathbf{x})),$$

$$\mathbf{R}_\phi\mathbf{x} = (x\cos\phi + y\sin\phi, y\cos\phi - x\sin\phi) \longrightarrow (\rho(\mathbf{x}), \phi + \theta(\mathbf{x})).$$



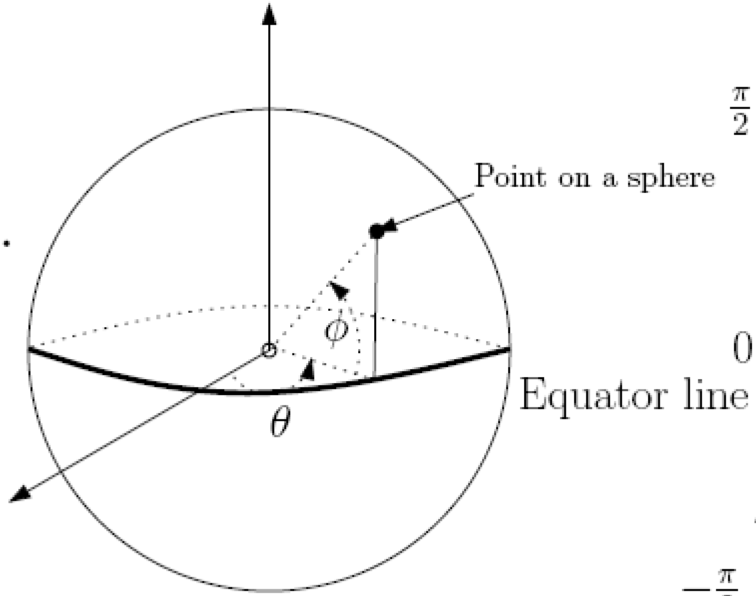


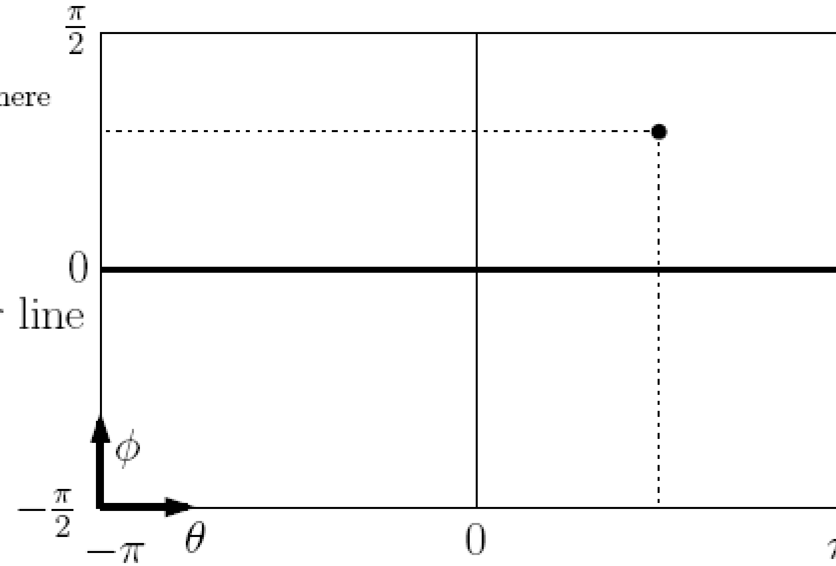
Data reduction for retinal images...

Spherical coordinates

$$\theta = \arctan \frac{x}{z} \quad \text{and} \quad \phi = \arctan \frac{y}{\sqrt{x^2+z^2}}.$$

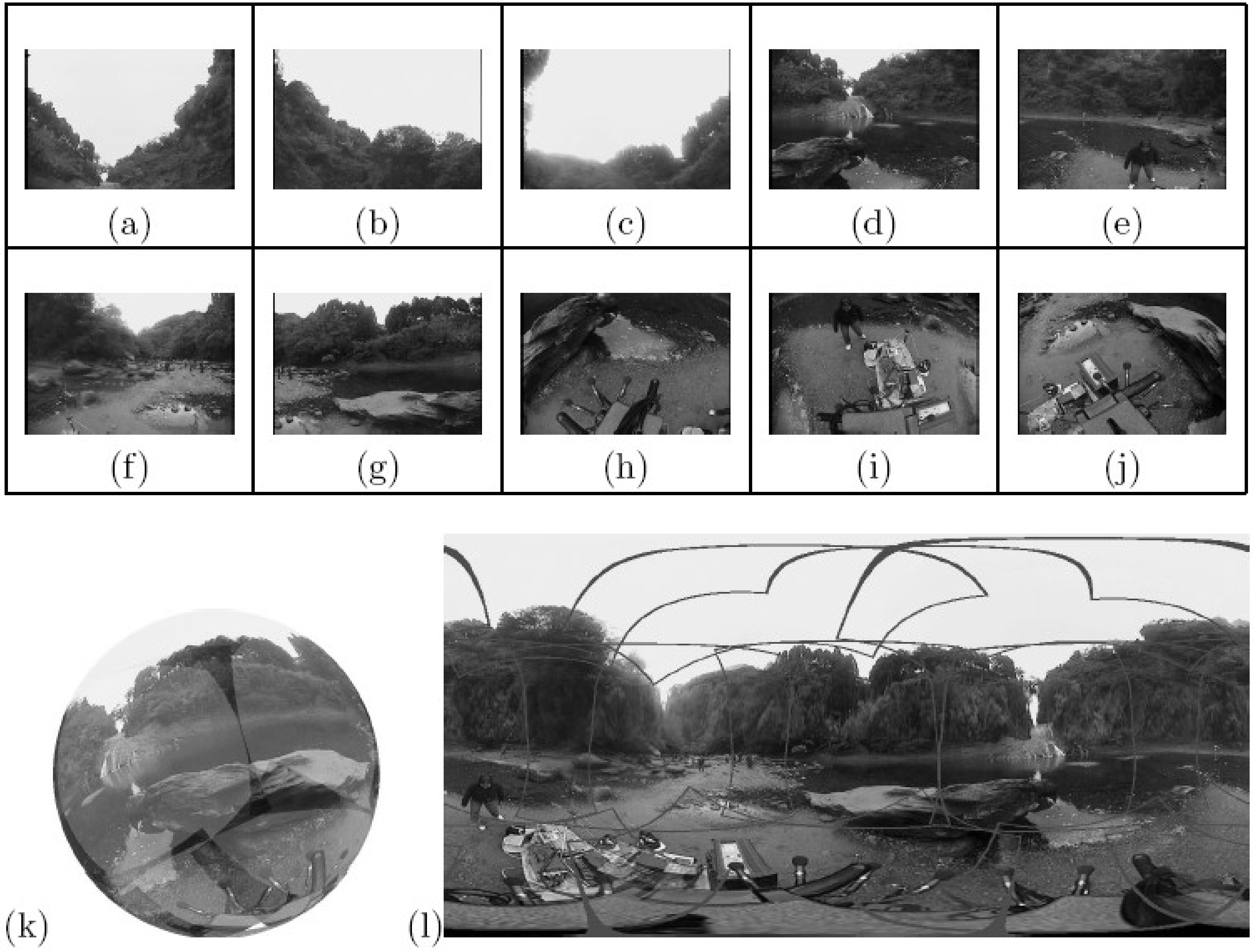
$$\mathbf{r} = \begin{bmatrix} \cos\phi \sin\theta \\ \sin\phi \\ \cos\phi\cos\theta \end{bmatrix}.$$





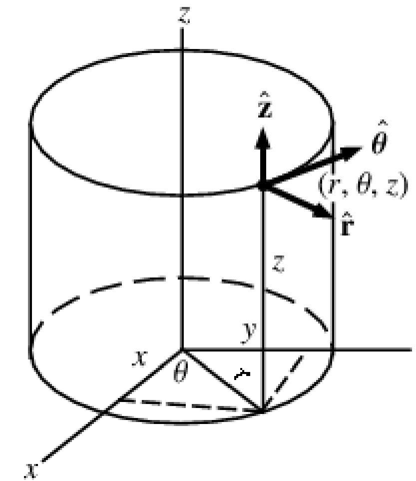
Spherical coordinates

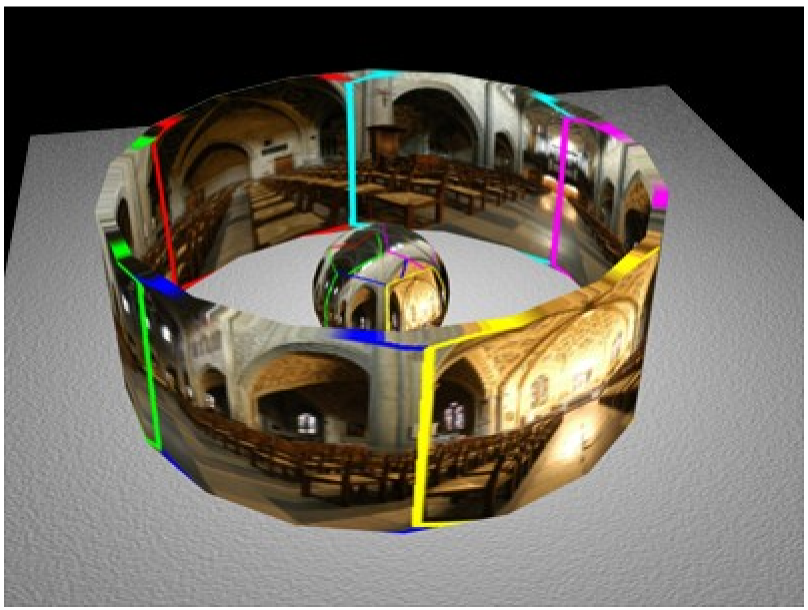
High Resolution Full Spherical Videos
Frank Nielsen, ITCC'02



Cylindrical coordinates

$$\theta = \arctan \frac{x}{z}, \qquad s = \frac{y}{\sqrt{x^2 + z^2}}.$$

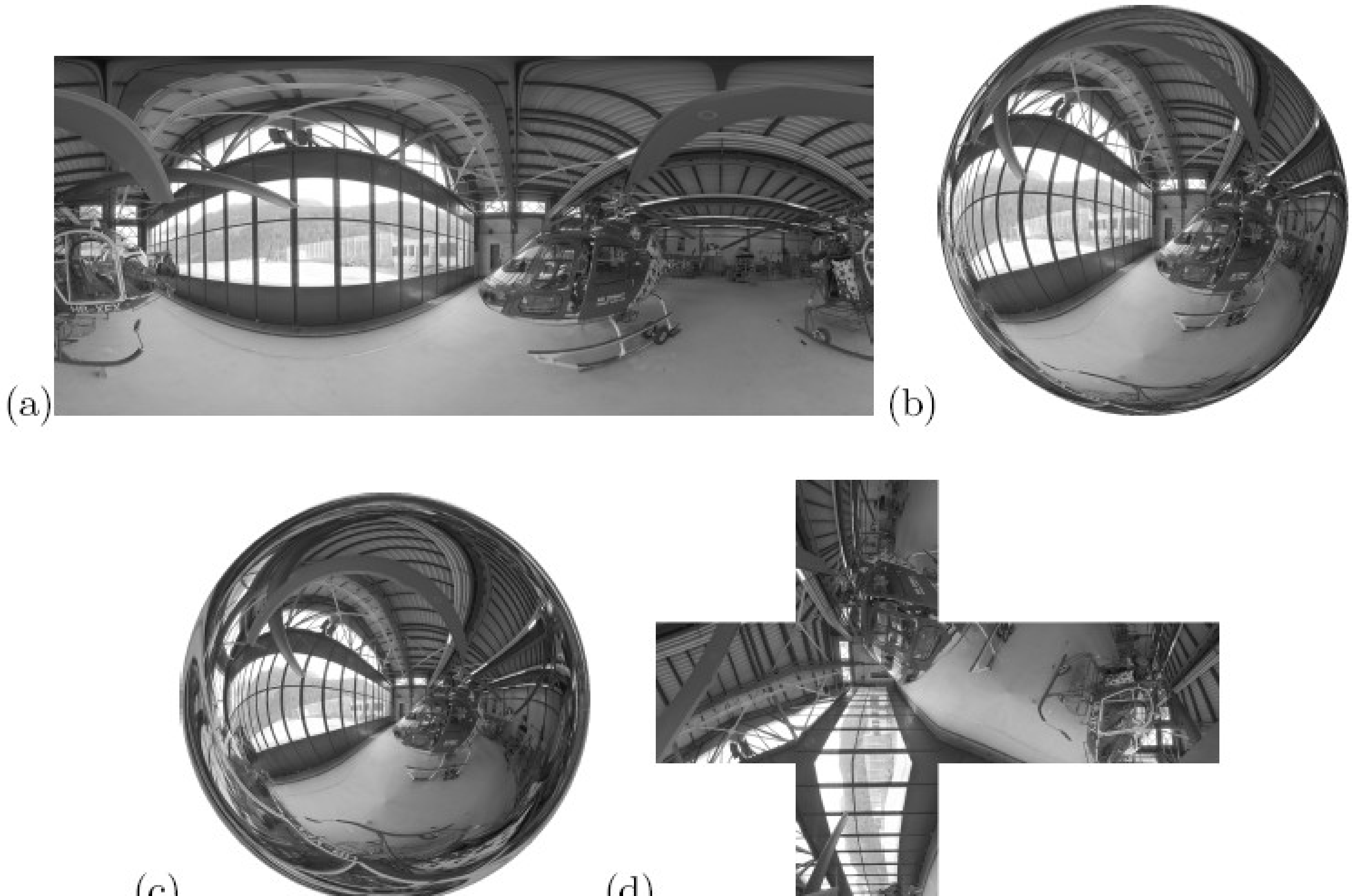




Panoramic image stitching:

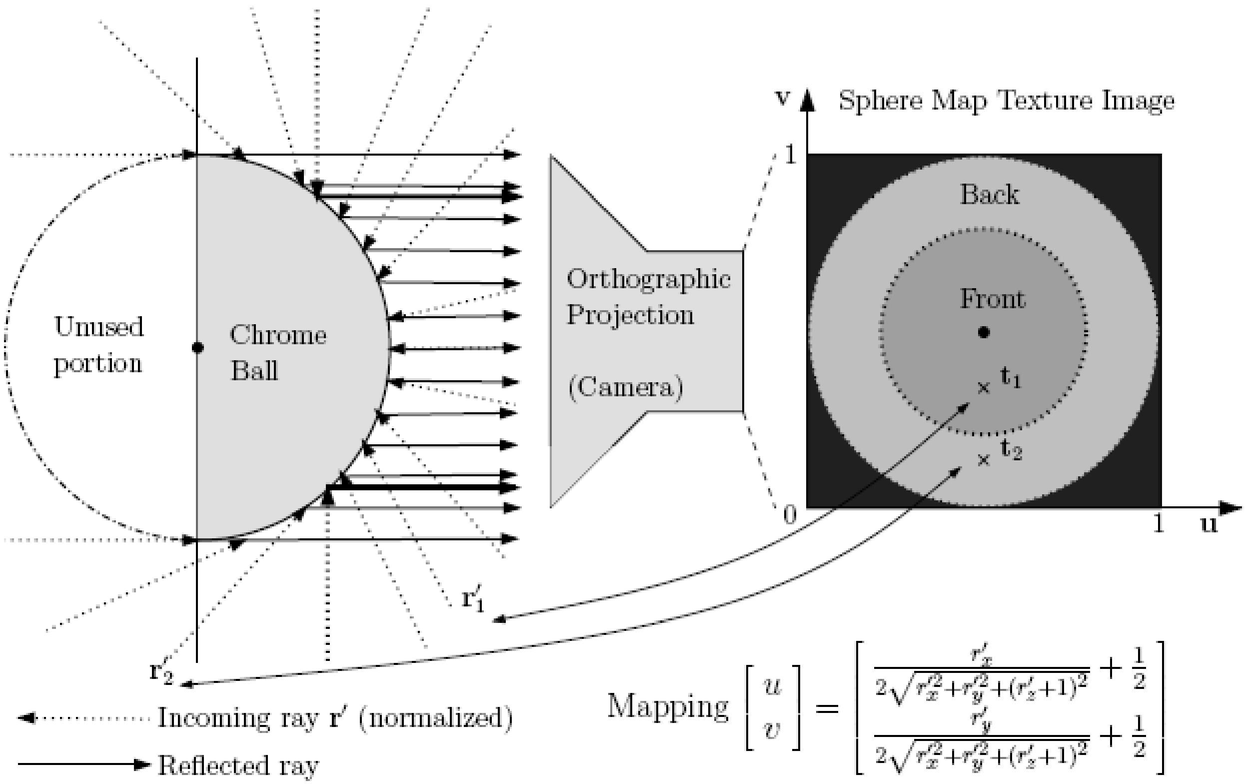
Align by a translation into the cylindrical coordinate map...

Environment maps



Equirectangular, mirror ball, cubic, etc.

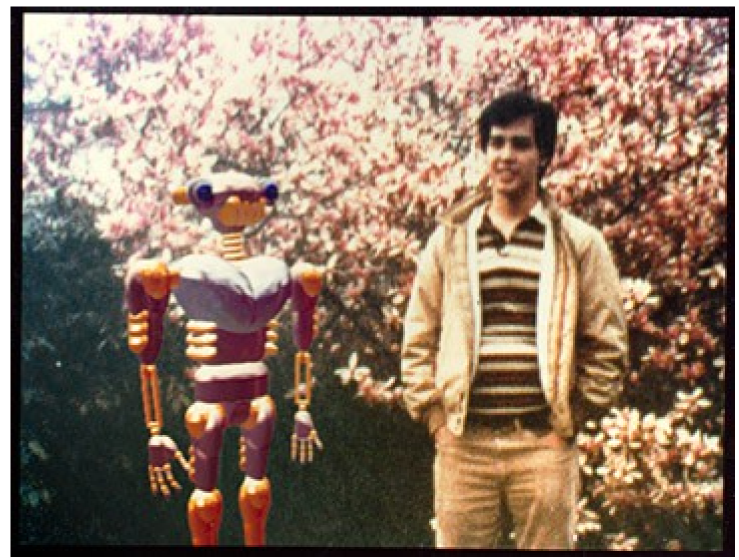
Environment maps: Mirror ball



Mapping $$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{r'_x}{2\sqrt{r'^2_x + r'^2_y + (r'_z+1)^2}} + \frac{1}{2} \\ \frac{r'_y}{2\sqrt{r'^2_x + r'^2_y + (r'_z+1)^2}} + \frac{1}{2} \end{bmatrix}$$

Environment maps for reflections

Blinn, 1976



1982

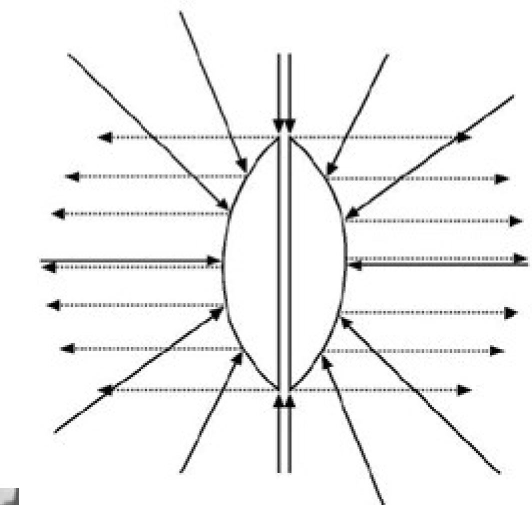
Interface, 1985
Lance Williams

http://www.debevec.org/ReflectionMapping/

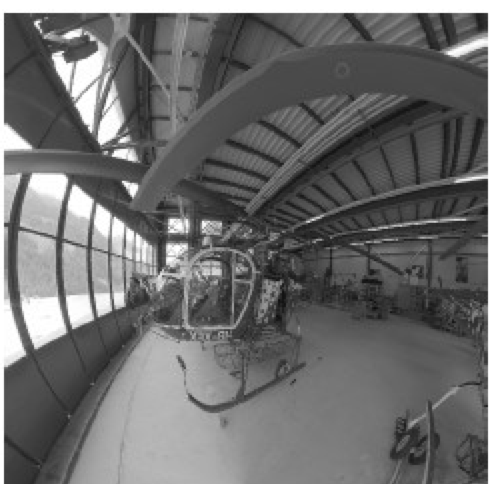
Environment maps for reflections

Abyss, Terminator 2 (1991)

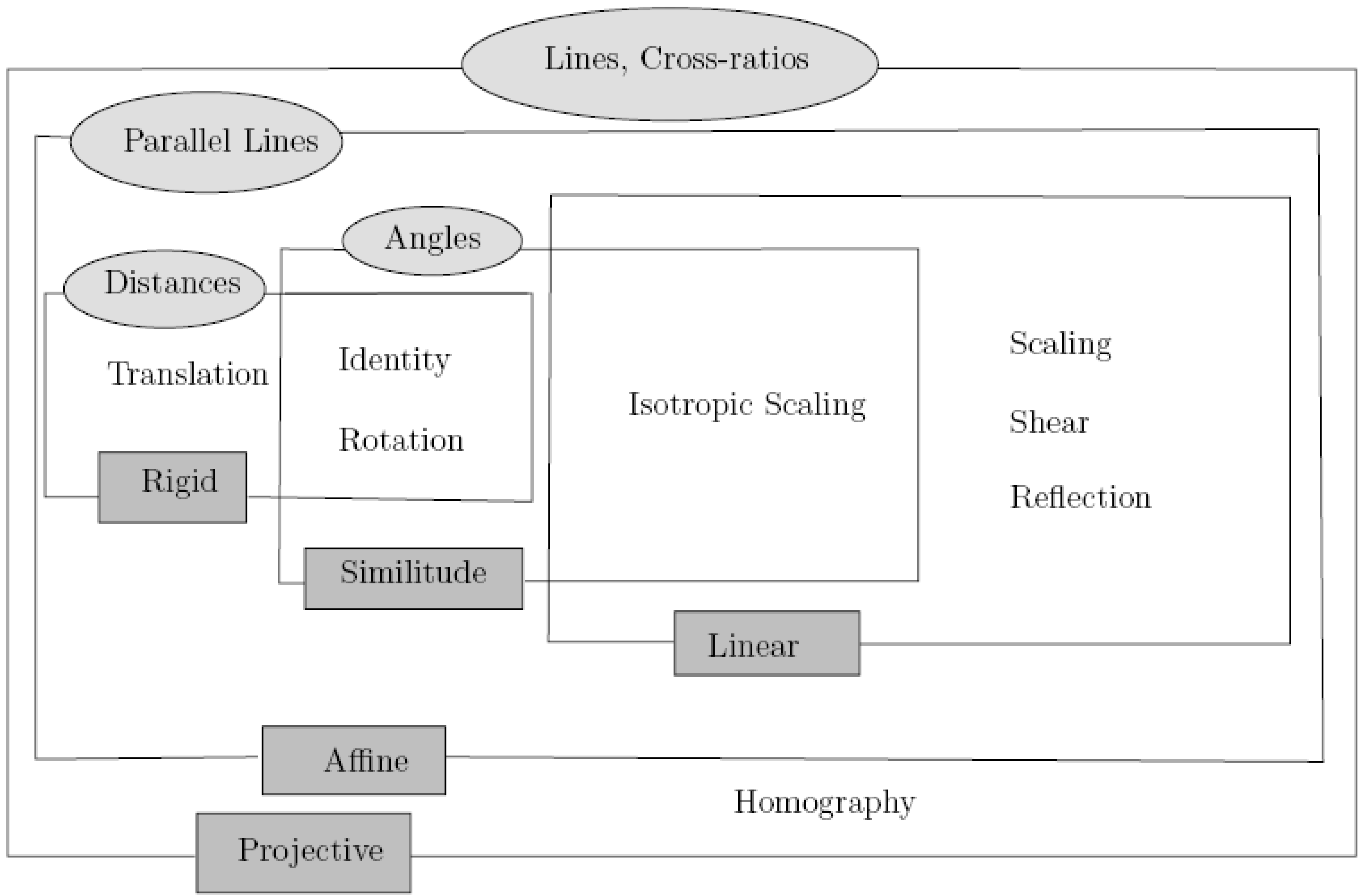
Best environment map for real-time graphics?



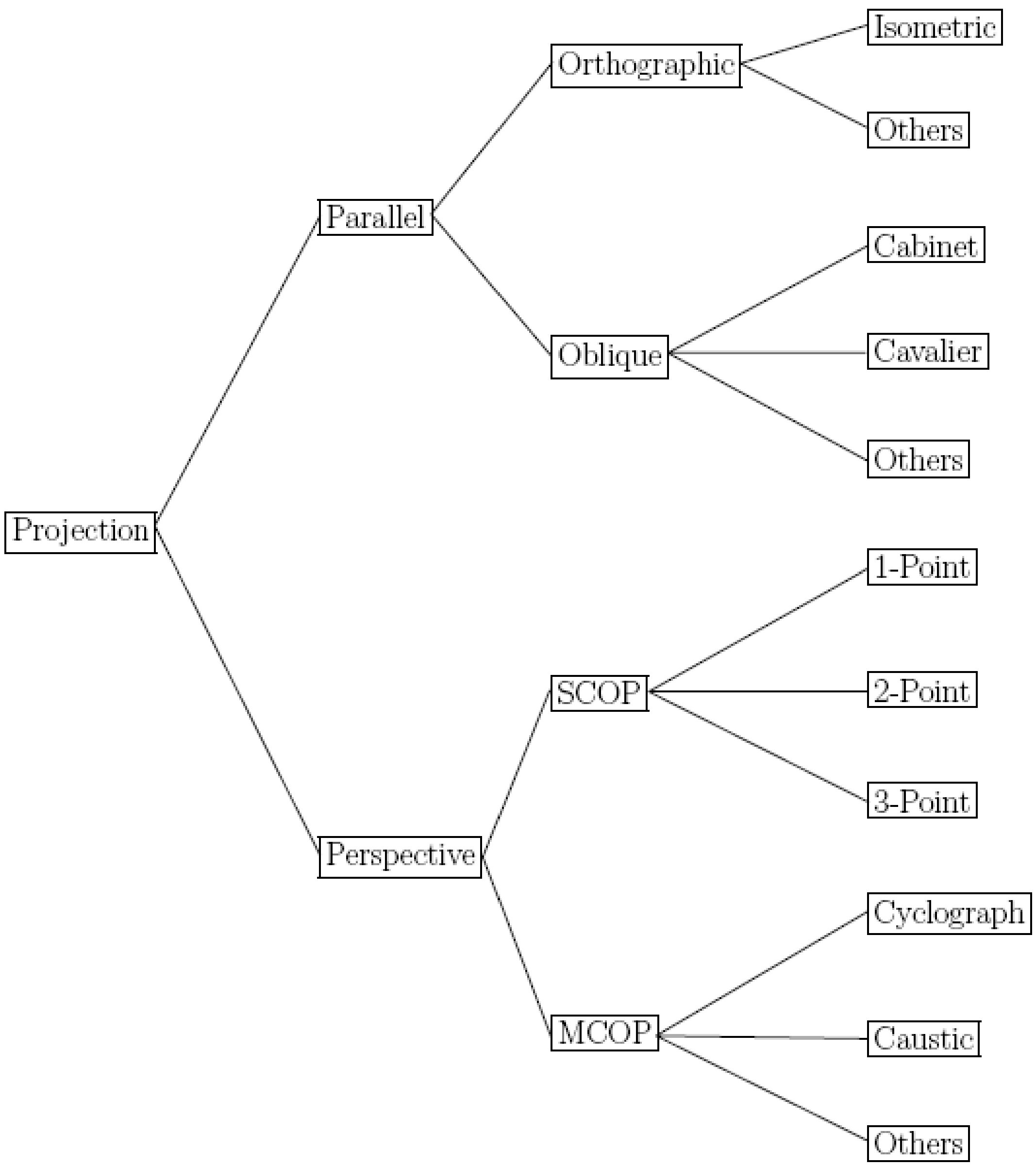
Dual paraboloid (2 images front/back only)



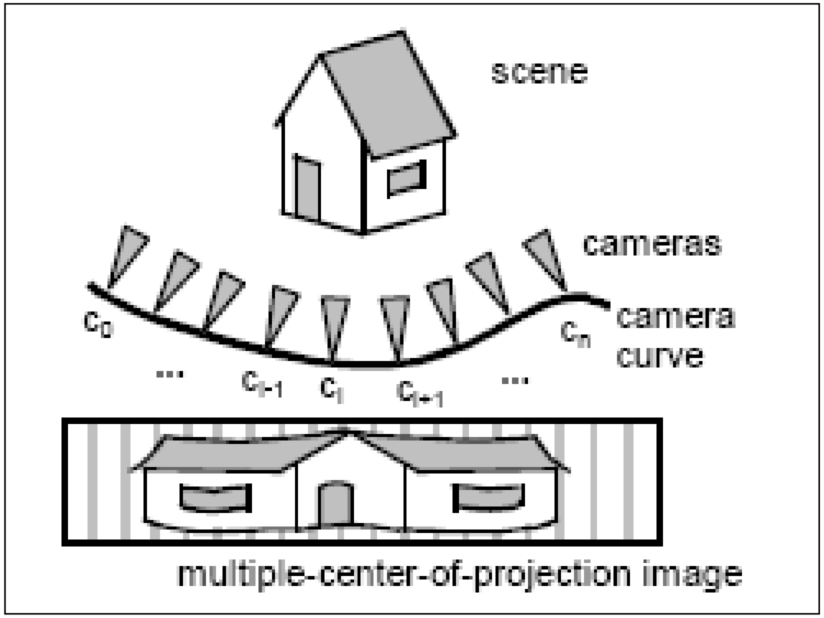
Transformations and their invariants

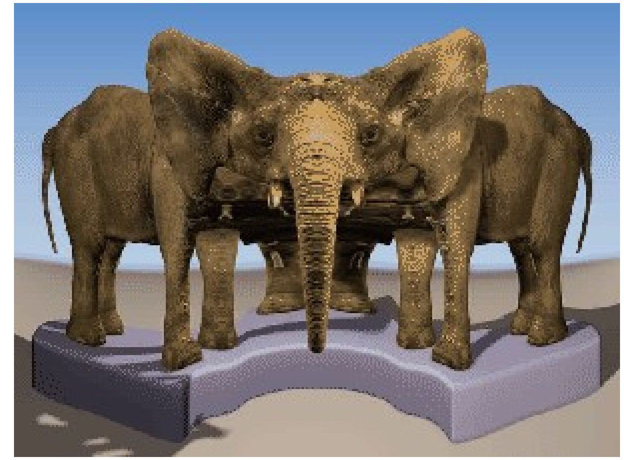


Taxonomy of projections:

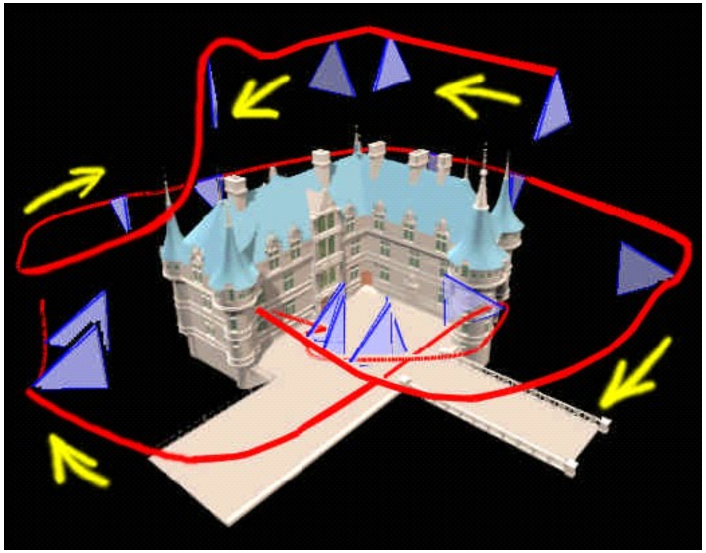


Multiple centers of projections (MCOP)

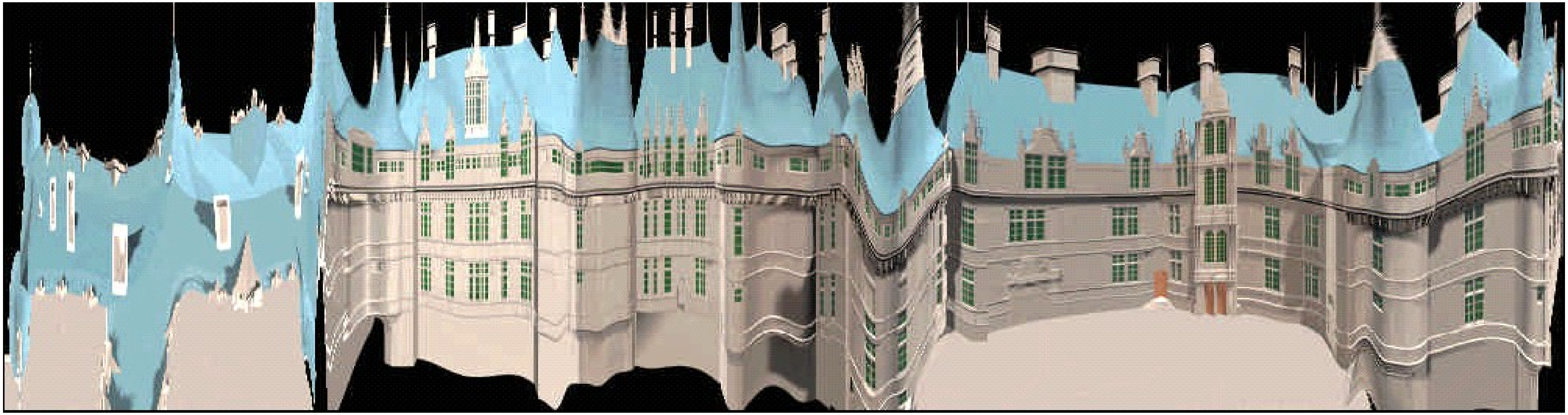


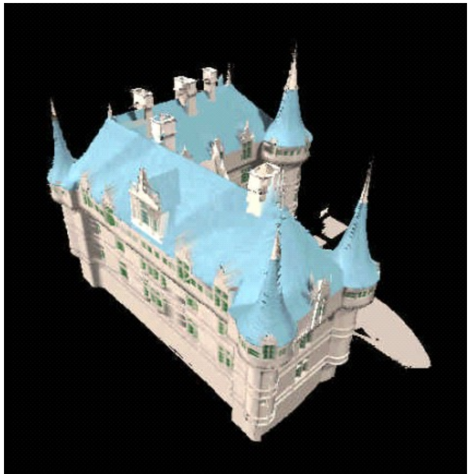


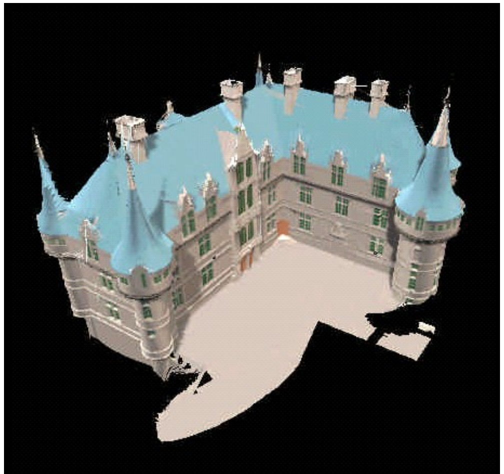
Reconstruction from a single MCOP images
Generalizes epipolar geometry
Resolution dependent



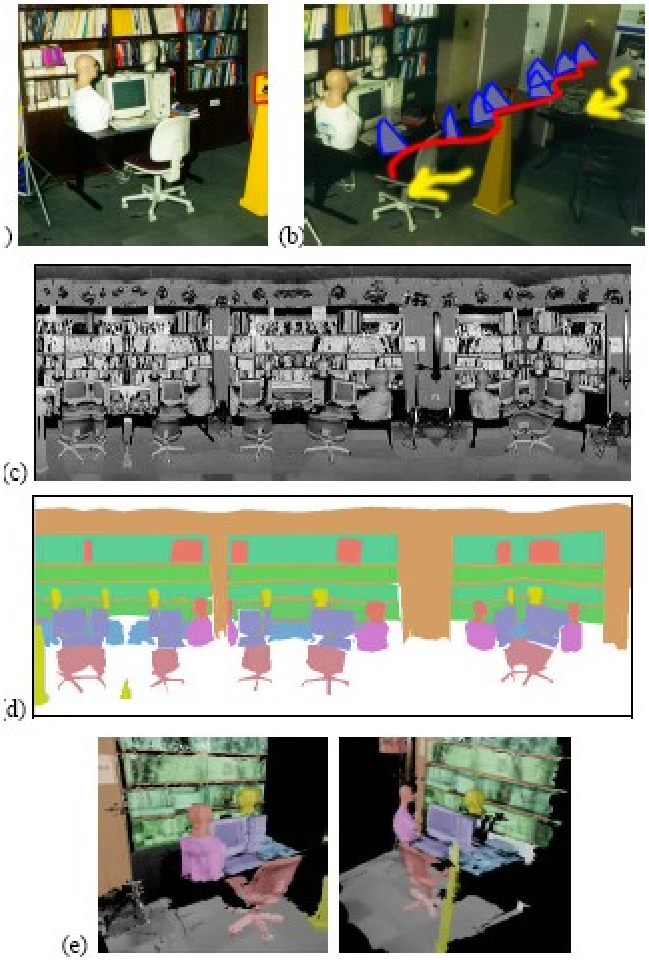
Acquisition Example





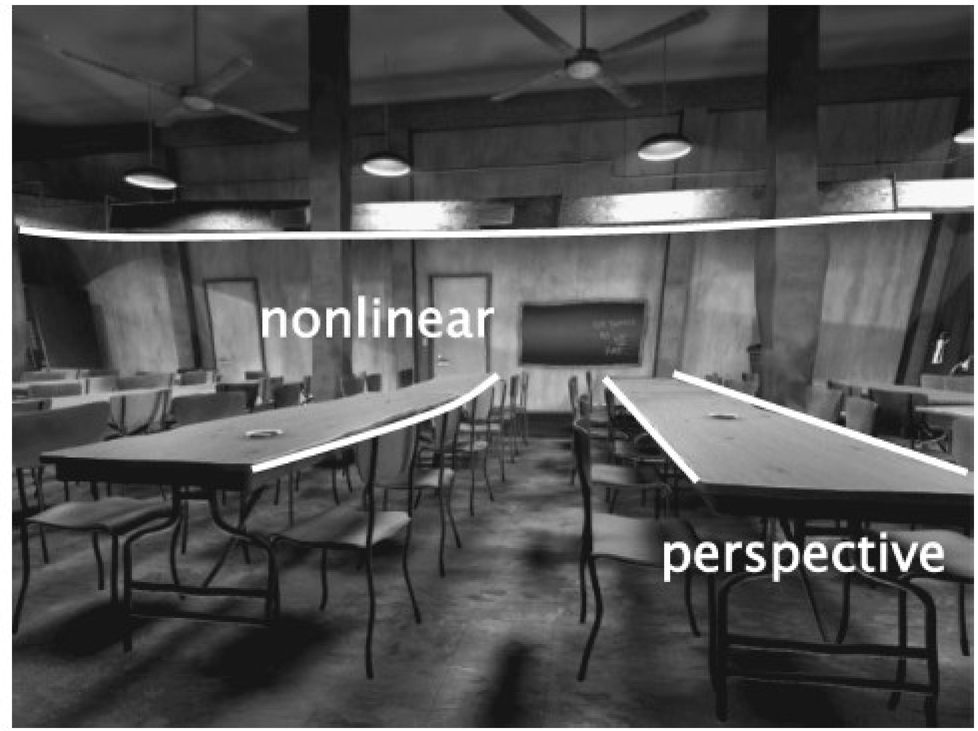


Multiple centers of projections (MCOP)



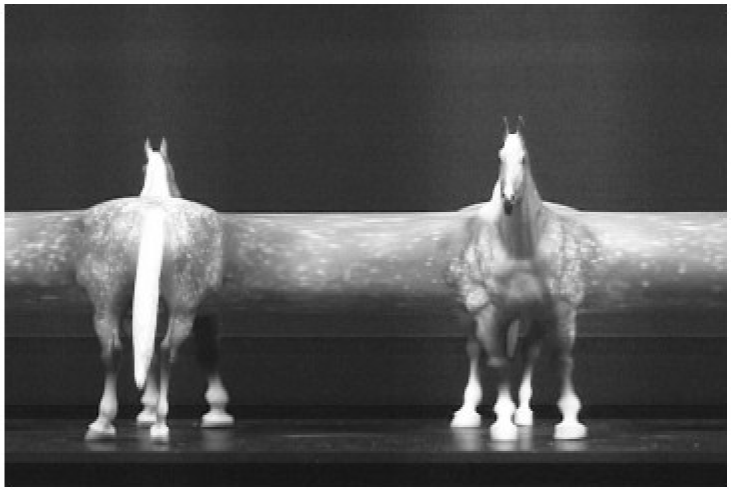
Difficult to obtain in practice...
Localization

Multiple centers of projections (MCOP)

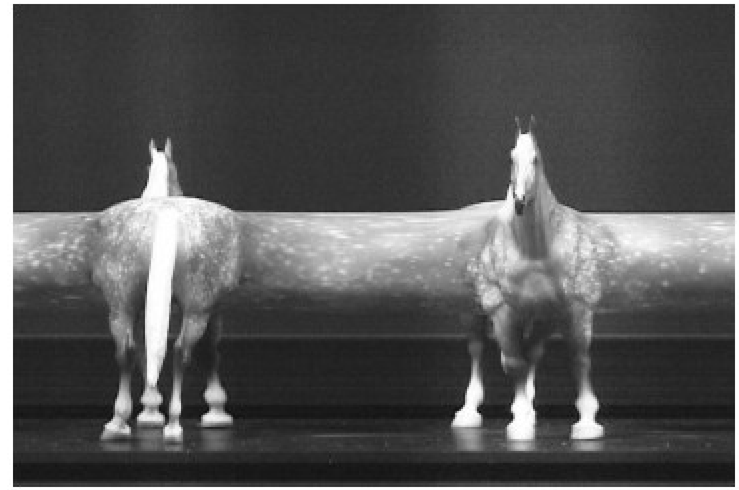


The art of depiction...

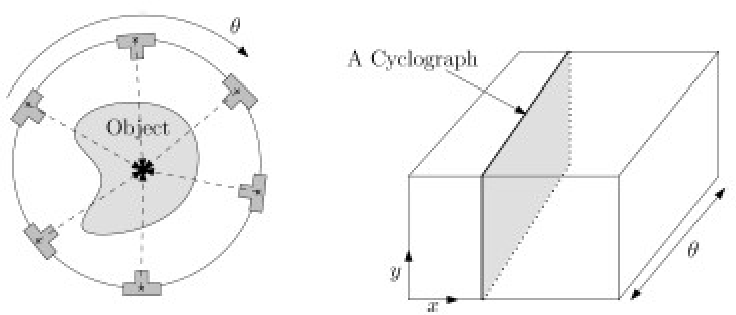
Stereo cyclographs...



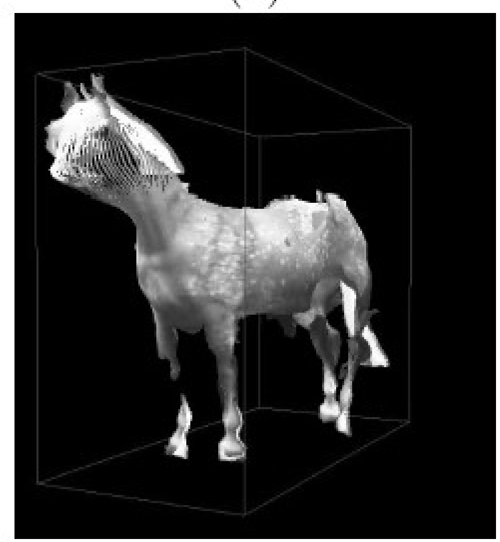
(a)



(b)

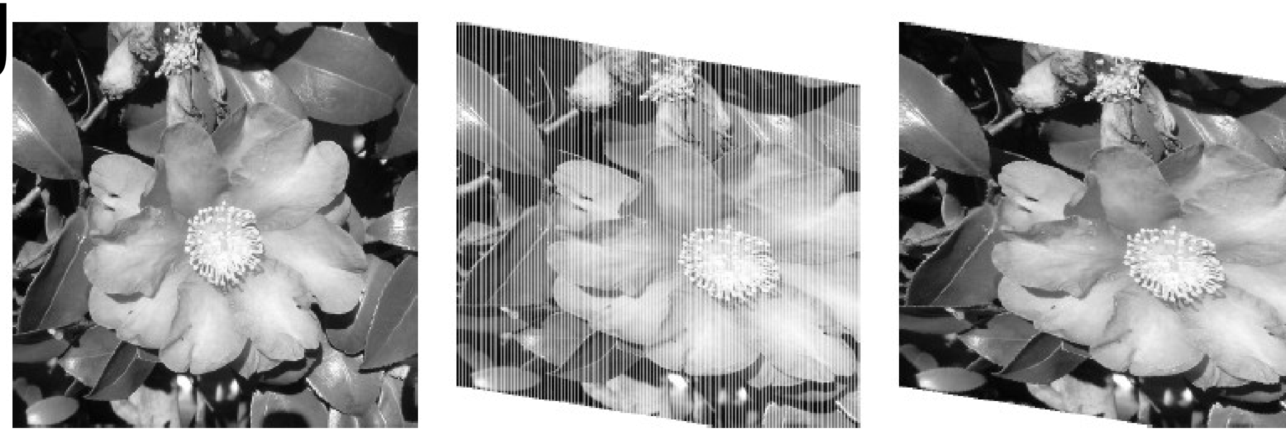


(c)



(d)

Image backward vs forward mapping
Image warping



```
FORWARDMAPPING(I_s, f)
◁ Create a warped image I_d by forward mapping ▷
◁ f: warping function ▷
Initialize an empty image I_d
◁ for all image lines ▷
for y ← 1 to h_s
    do ◁ for all column pixels ▷
       for x ← 1 to w_s
           do ◁ Compute the source-to-destination mapping ▷
              (u, v) ← f(x, y)
              ◁ Round coordinates to integers ▷
              ◁ (no interpolation required) ▷
              (u_r, v_r) ← (⌊u⌉, ⌊v⌉)
              ◁ Should check index bounds ▷
              I_d[u_r, v_r] = I_s[x, y]
```

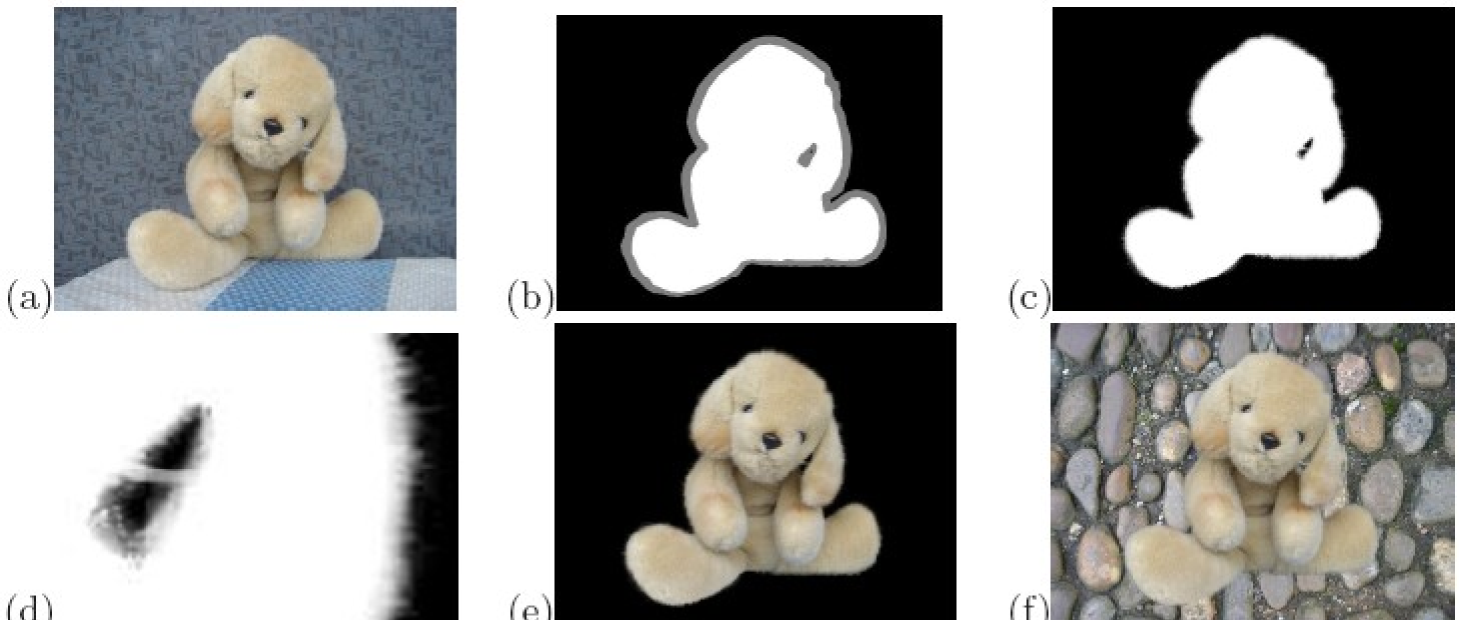
```
BACKWARDMAPPING(I_s, f)
◁ Destination image I_d of dimension w_d × h_d ▷
for v ← 1 to h_d
    do for u ← 1 to w_d
           do (x, y) = g(u, v)
              ◁ Backward mapping requires resampling ▷
              I_d[u, v] = RESAMPLE(I_s, x, y)
```

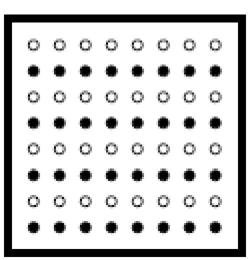
Resampling
Interpolation

Image Blending: Alpha channel

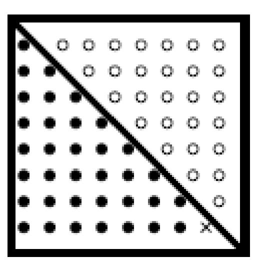
$$\mathbf{I}[i,j] = \boldsymbol{\alpha}[i,j]\mathbf{F}[i,j] + (1 - \boldsymbol{\alpha}[i,j])\mathbf{B}[i,j],$$

$$\mathbf{I} = \alpha\mathbf{F} + (1-\alpha)\mathbf{B},$$

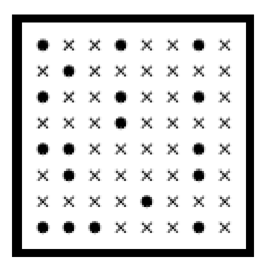




Full semitransparent coverage ($\alpha = \frac{1}{2}$)



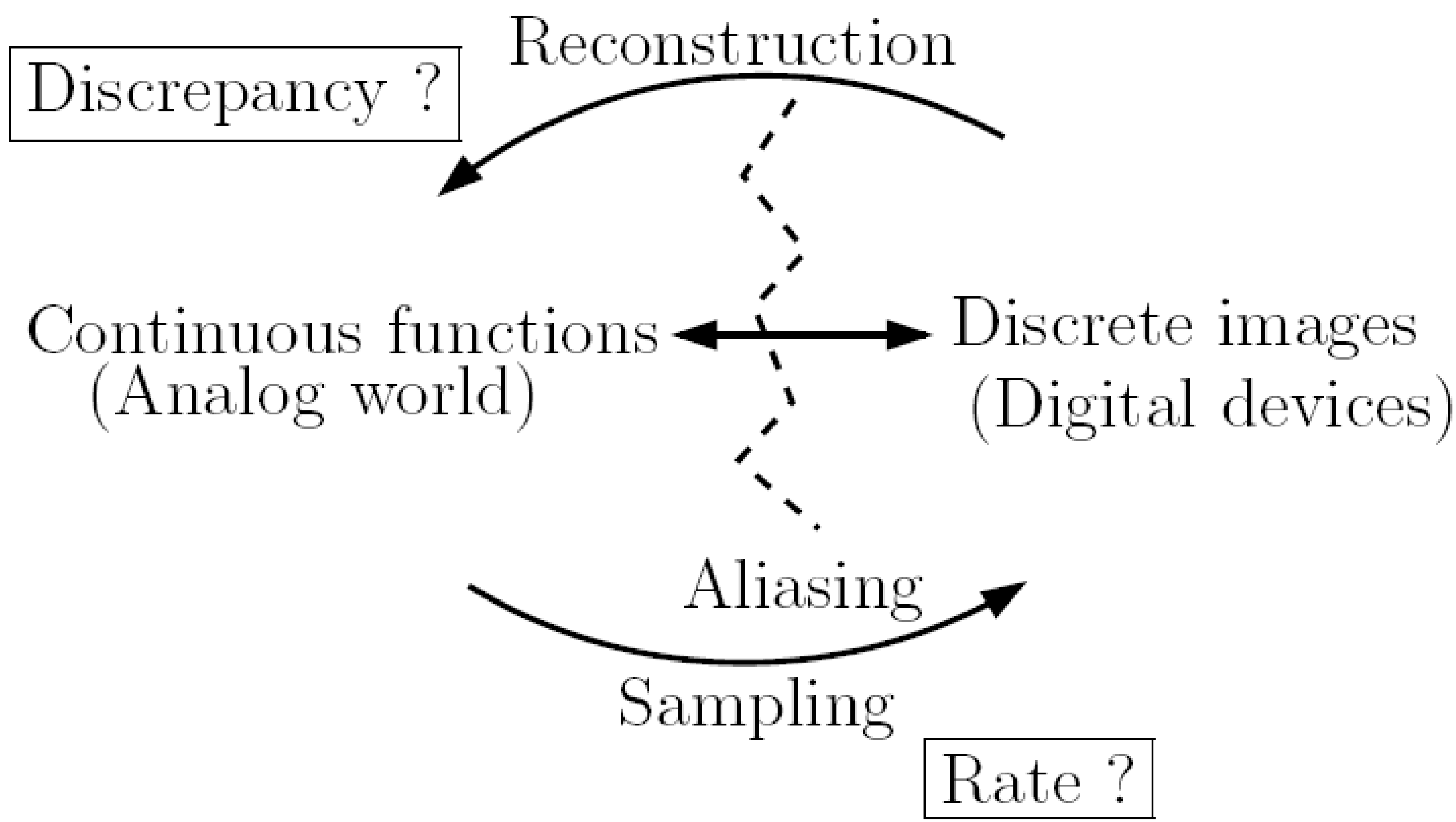
Partial opaque coverage ($\alpha = \frac{1}{2}$)



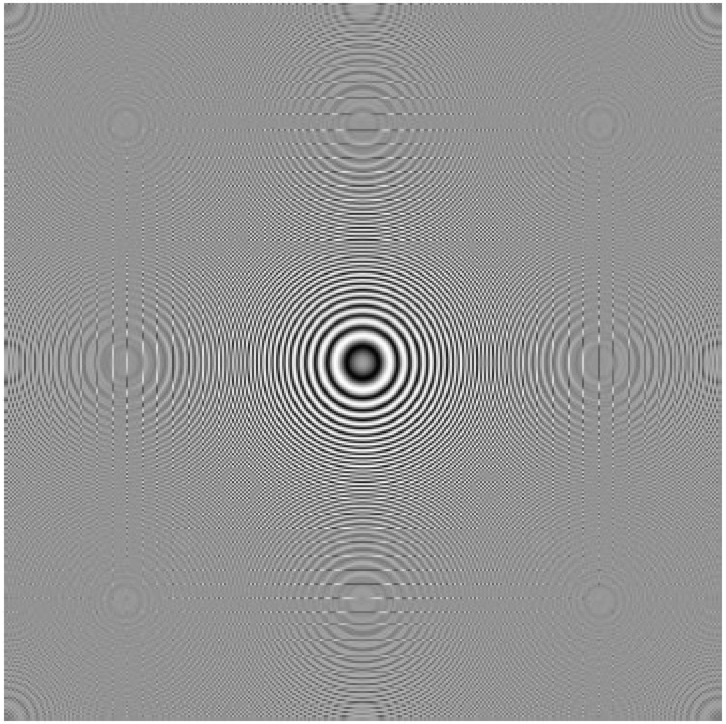
Blending two colors • ×

Interpretation at the microscopic level...

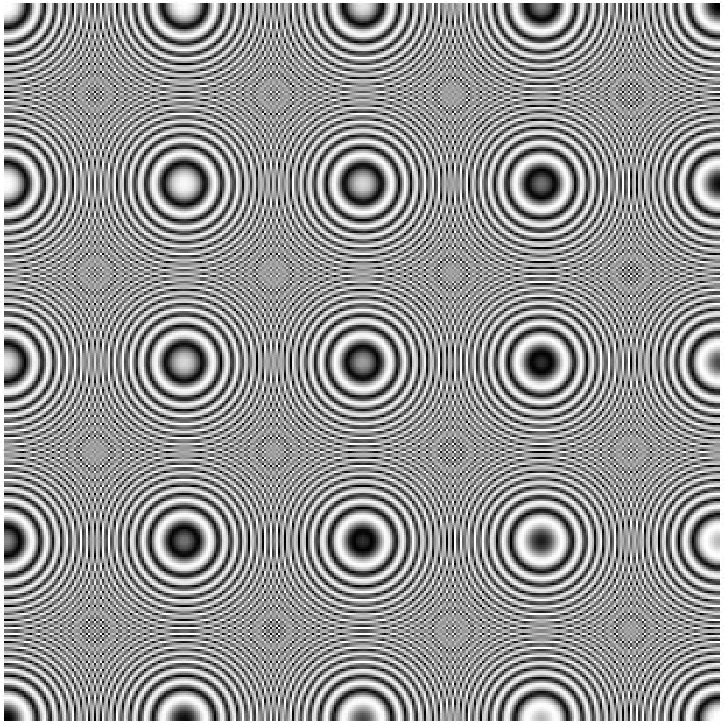
Image sampling/reconstruction



Zone plate: Aliasing/ringing effect



1024x1024



256x256
downsampled
using bilinear interpolation

Continuous versus discrete convolutions

$$(f \otimes g)(x) = \int_{t=-\infty}^{\infty} f(t)g(x-t)dt = \int_{t=-\infty}^{\infty} g(t)f(x-t)dt = (g \otimes f)(x).$$

$$\mathbf{C}[i,j] = \mathbf{A} \otimes \mathbf{B} = \sum_k \sum_l \mathbf{A}[k,l]\mathbf{B}[i-k,j-l].$$

$$\mathbf{G} = \frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Fourier analysis

$$f(x+T) = f(x)$$

Fourier discovered that ***all periodic signals*** can be represented as a sum (eventually infinite) of sinusoidal waves: sin(·) functions, the basis functions.

Let *f*(·) denote the continuous function in the spatial domain and *F*(·) denote the dual complex function, also called *spectral function*.

Euler formula (period 2pi)

$$\exp(ix) = \cos x + i \sin x$$

$$\exp(ix) = \cos x + i \sin x = \cos(x+2\pi) + i \sin(x+2\pi) = \exp(i(x+2\pi))$$

Fourier analysis: Duality spatial/spectral domain

Spatial domain

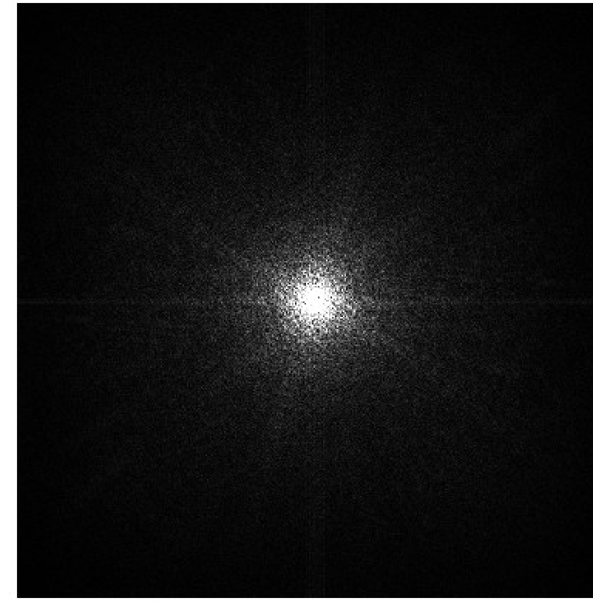
$$f(x,y) = \int_{u=-\infty}^{\infty} \int_{v=-\infty}^{\infty} F(u,v) \exp\left(i2\pi(ux+vy)\right) du dv.$$

Spectral domain

$$F(u,v) = \int_{x=-\infty}^{\infty} \int_{y=-\infty}^{\infty} f(x,y) \exp\left(-i2\pi(ux+vy)\right) dx dy.$$

$$F(u,v) = A(u,v) + iB(u,v)$$

Fourier analysis: Phase/amplitude



Frequency magnitudes

$$F(u,v) = A(u,v) + iB(u,v)$$

Polar coordinates:

$$P(u,v) = A^2(u,v) + B^2(u,v)$$

$$F(u,v) = |F(u,v)| \exp(i\phi(u,v))$$

Power spectrum

$$|F(u,v)|$$

$$\phi(u,v) = \arctan \frac{B(u,v)}{A(u,v)}$$

Amplitude

Phase

Fourier analysis: Convolution theorem

Convolution in spatial domain is a multiplication in Fourier domain

$$\mathcal{F}(f \otimes g) = \sqrt{2\pi}(\mathcal{F}f) \times (\mathcal{F}g) = \sqrt{2\pi}F \times G$$

Convolution in frequency domain is a multiplication in spatial domain

$$F \otimes G = \sqrt{2\pi}\mathcal{F}(f \times g)$$

Fourier analysis: Discrete transformations

1D

$$f_j = \frac{1}{n}\sum_{k=0}^{n-1} x_k \exp(-2\pi i \frac{jk}{n}), \;\forall\; 0 \leq j \leq n-1$$

$$x_k = \sum_{j=0}^{n-1} f_j \exp(2\pi i \frac{jk}{n})$$

2D

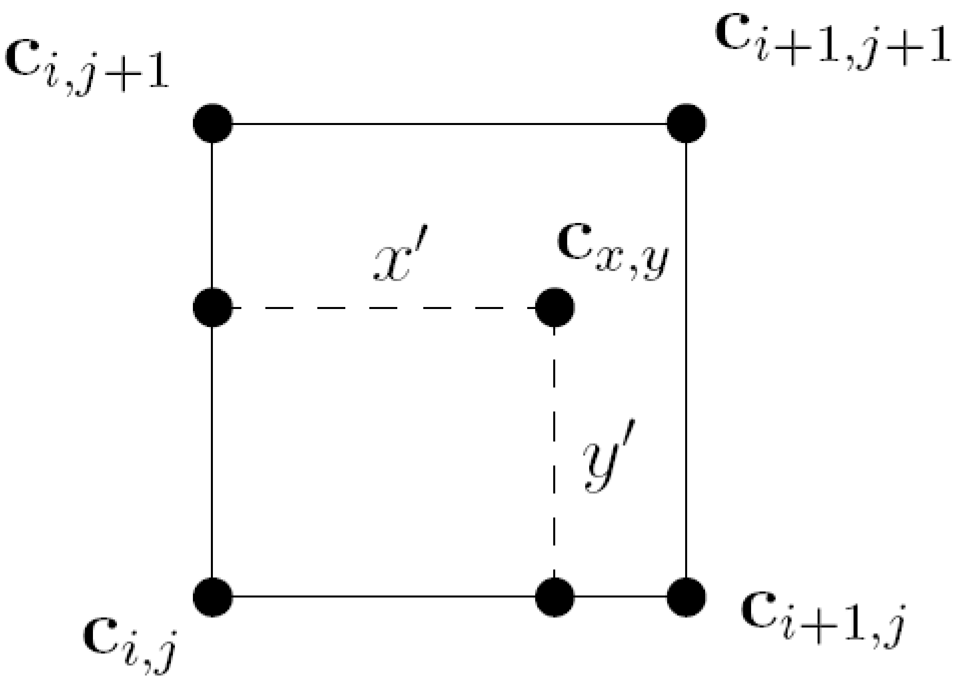
$$F(u,v) = \frac{1}{wh}\sum_{x=0}^{w-1}\sum_{y=0}^{h-1} f(x,y) \exp\left(-2\pi i \left(\frac{xu}{w} + \frac{yv}{h}\right)\right)$$

$$f(x,y) = \sum_{u=0}^{w-1}\sum_{v=0}^{h-1} F(u,v) \exp\left(2\pi i \left(\frac{xu}{w} + \frac{yv}{h}\right)\right)$$

Interpolation/reconstruction filters

Bilinear interpolation

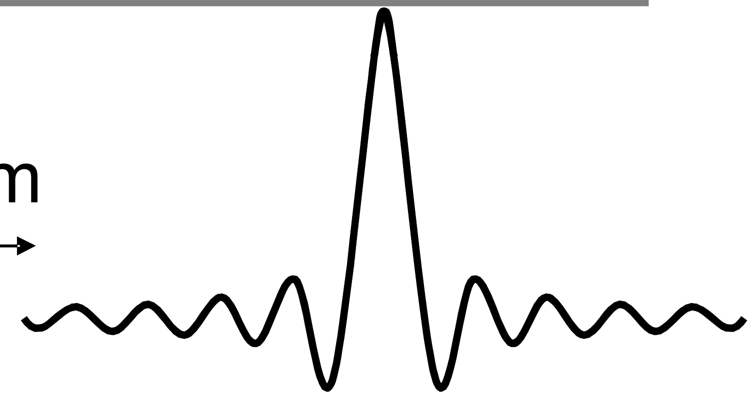
$$\mathbf{c}_{x,y} = (1-x')y'\mathbf{c}_{i,j+1} + x'y'\mathbf{c}_{i+1,j+1} + (1-x')(1-y')\mathbf{c}_{i,j} + x'(1-y')\mathbf{c}_{i+1,j}$$



Interpolation/reconstruction filters

Sinc (Lanczos) is ideal low-pass filter (infinite support)

Fourier transform



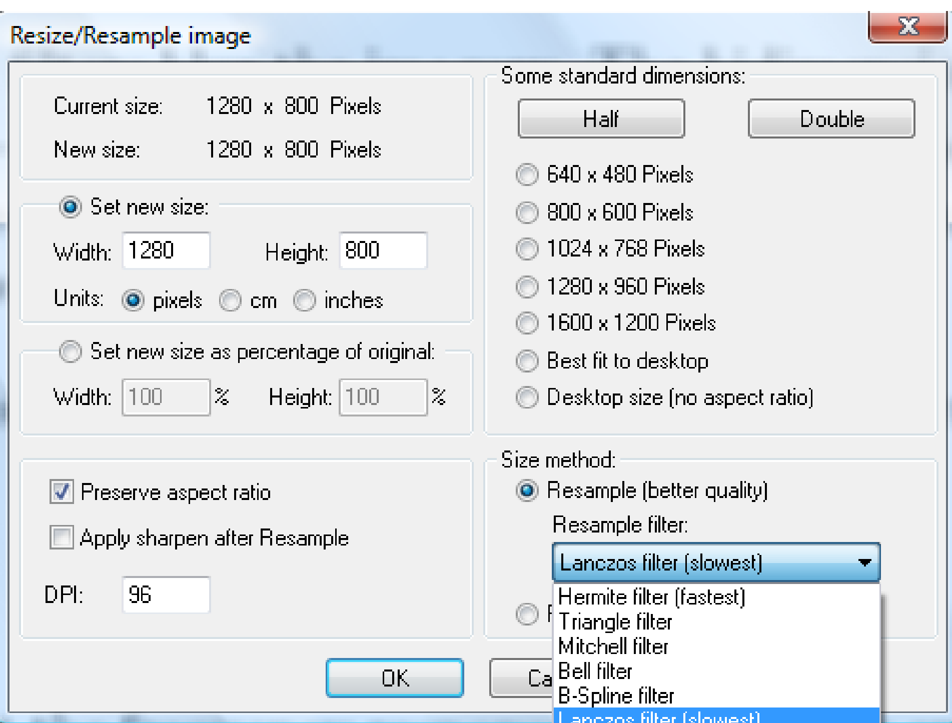
Ideal low-pass filter

The sinc function

$$\mathrm{sinc}(x) = \frac{\sin \pi x}{\pi x}$$

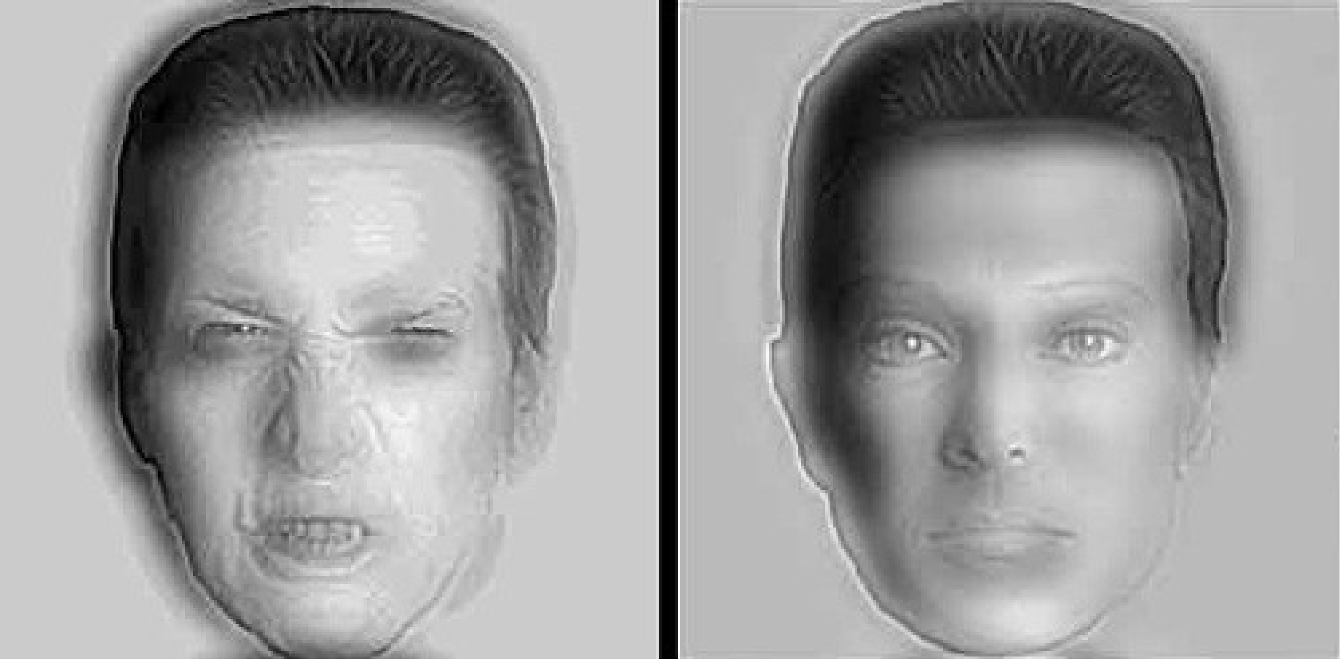
Fourier domain

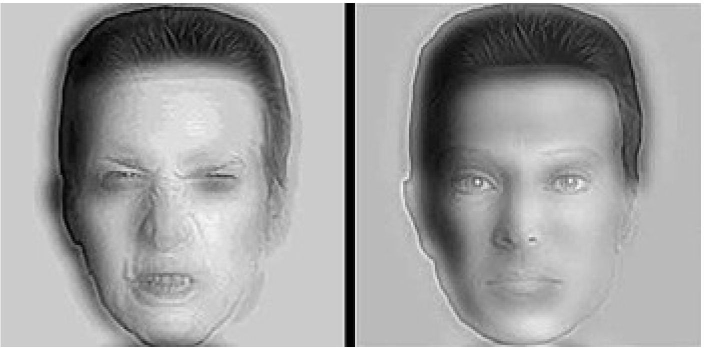
Spatial domain



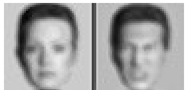
Windowed sinc

Mr Angry
Mrs Calm





Low/high frequency perception



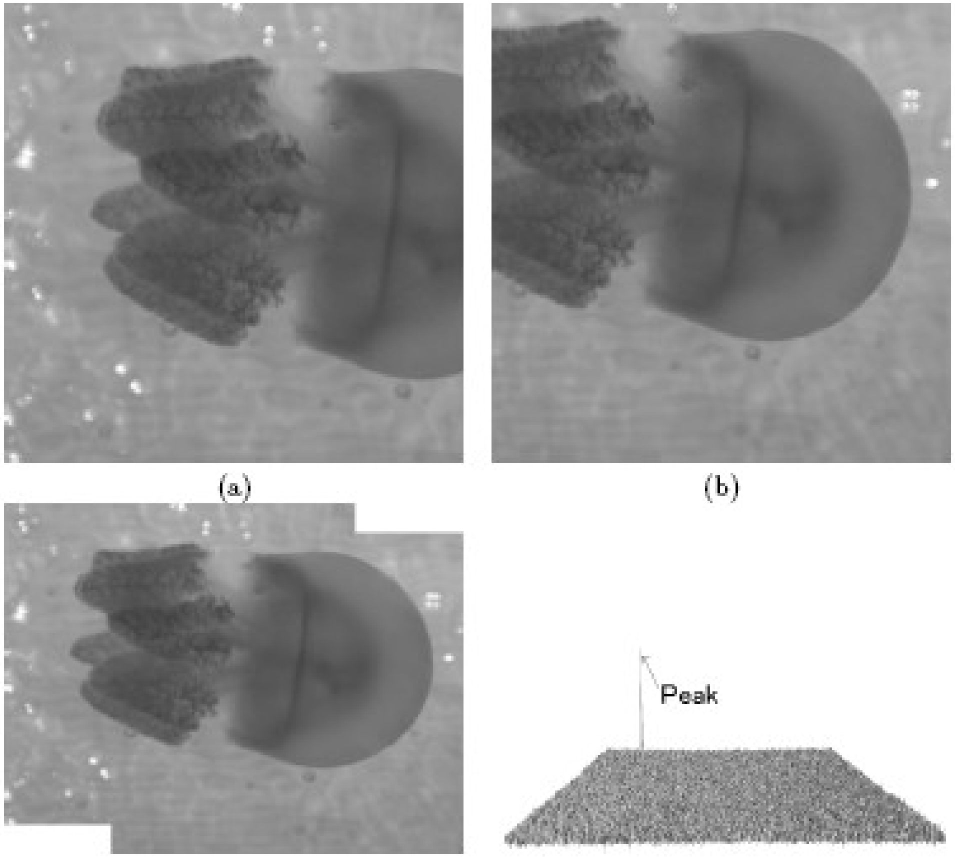
Phase correlation

Stitch by 2D translation two images

$$f_2(x,y) = f_1(x + x_t, y + y_t)$$

$$F_2(u,v) = F_1(u,v)\exp(-2\pi i(ux_t + vy_t))$$

$$\underbrace{\frac{F_1(u,v)F_2^*(u,v)}{|F_1(u,v)F_2^*(u,v)|}}_{\text{Cross-power spectrum}} = \exp(2\pi i(ux_t + vy_t))$$

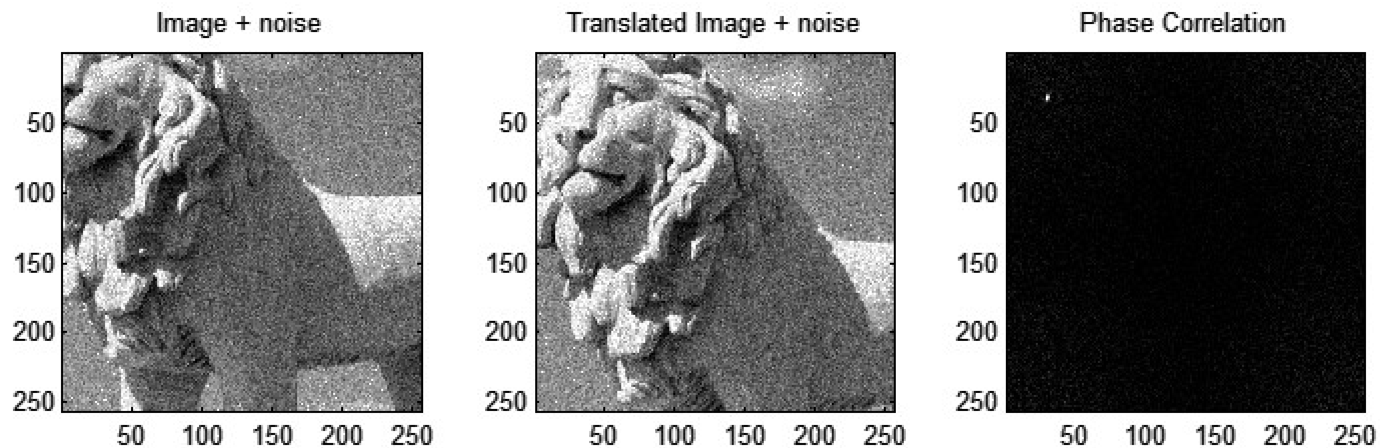


FFT can be computed in O(nlog n) time

F* is the conjugate function

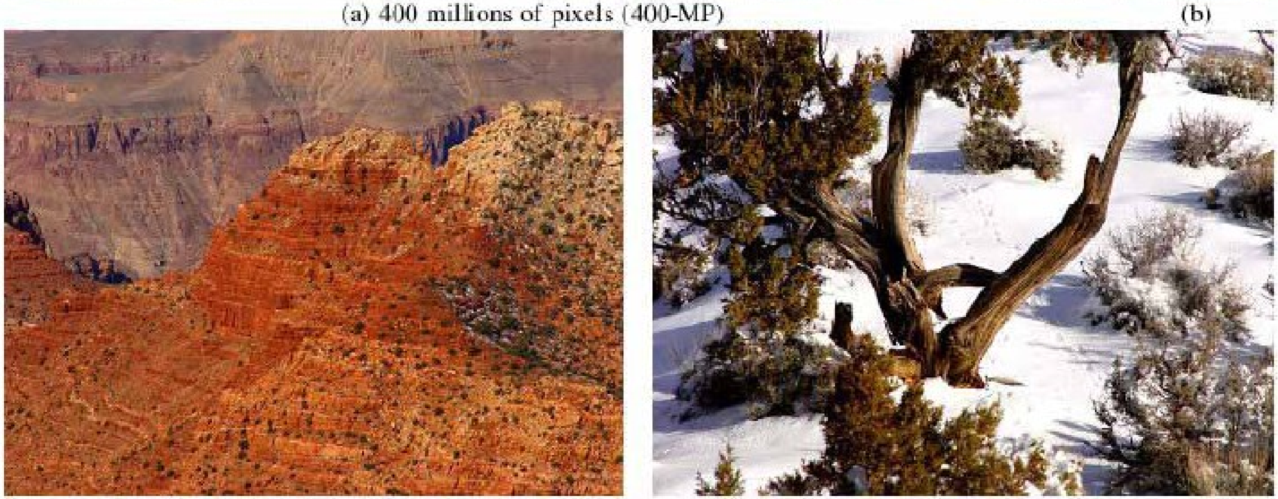
Phase correlation: Detecting the peak

Sub-pixel accuracy if we fit a quadratic function



(a) 400 millions of pixels (400-MP)

(b)



Clairvoyance system