COMP9517: Computer Vision

Image Formation

Image Formation

- « Image formation occurs when a **sensor** registers **radiation** that has interacted with **physical objects** » Ballard & Brown

scene

?

Geometry of image formation

Mapping **world coordinates to image coordinates**

- Pinhole camera model
- Projective geometry
- Projection matrix

Image formation

Idea 1: Put a piece of film in front of an object

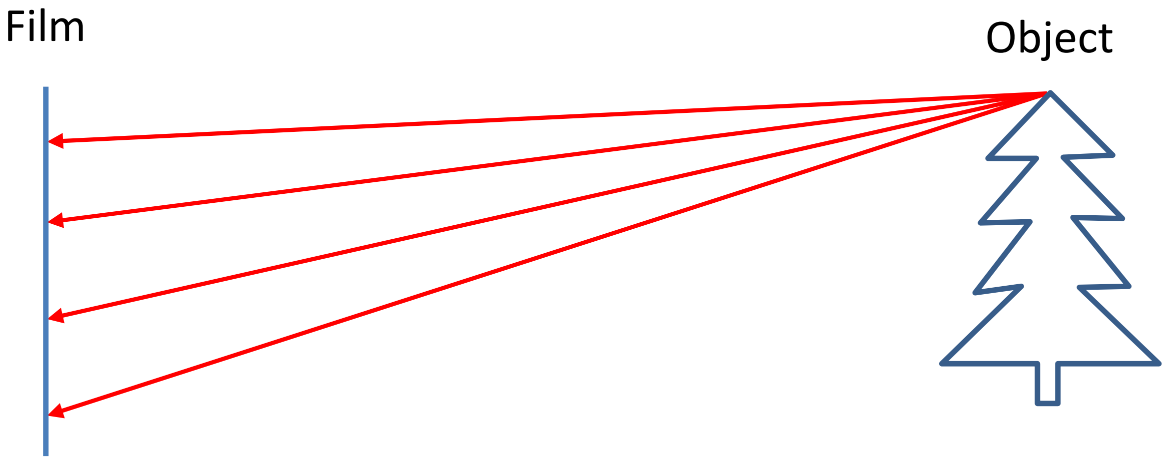
Do we get a reasonable image?

Image formation

Idea 1: Put a piece of film in front of an object

Do we get a reasonable image?

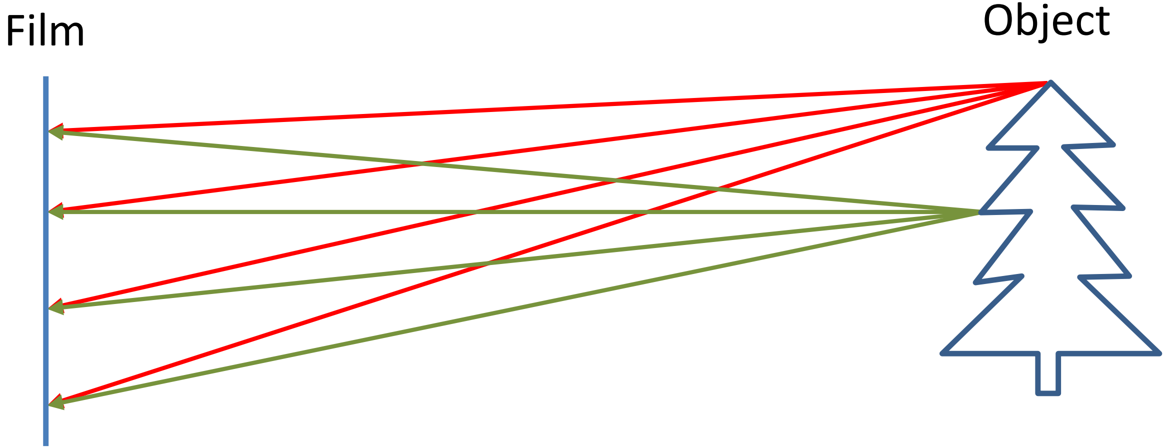
Image formation



Idea 1: Put a piece of film in front of an object

Do we get a reasonable image?

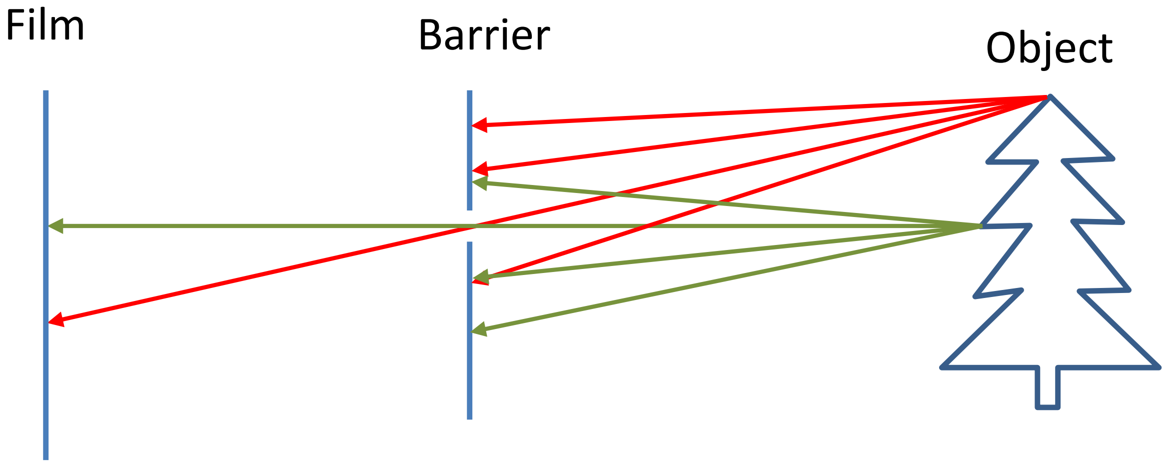
Image formation



Idea 1: Put a piece of film in front of an object

Do we get a reasonable image?

Image formation

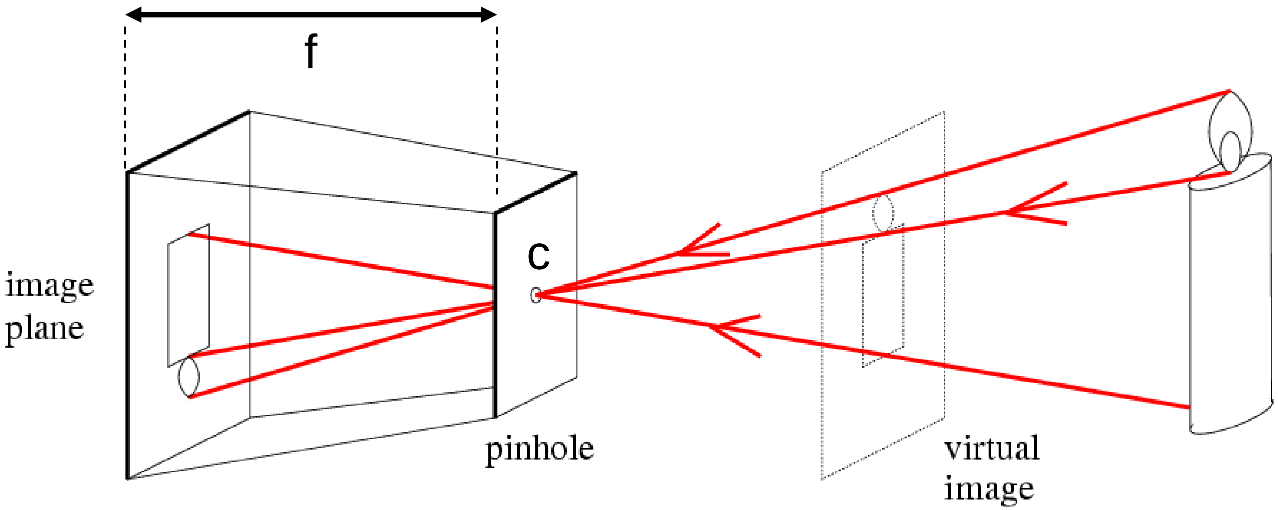


Idea 2: Add a barrier to block off most of the rays

This reduces blurring significantly

Opening known as the **pinhole** or **aperture**

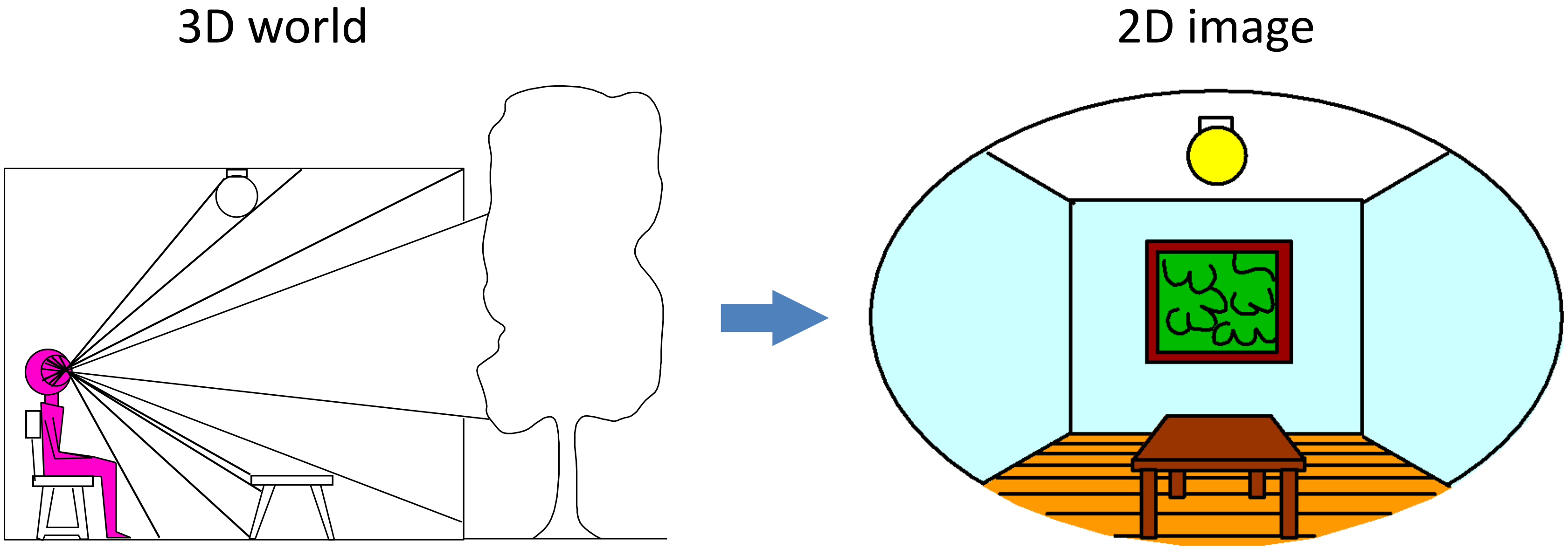
Pinhole camera model



f = focal length
c = centre of the camera

Figure from Forsyth

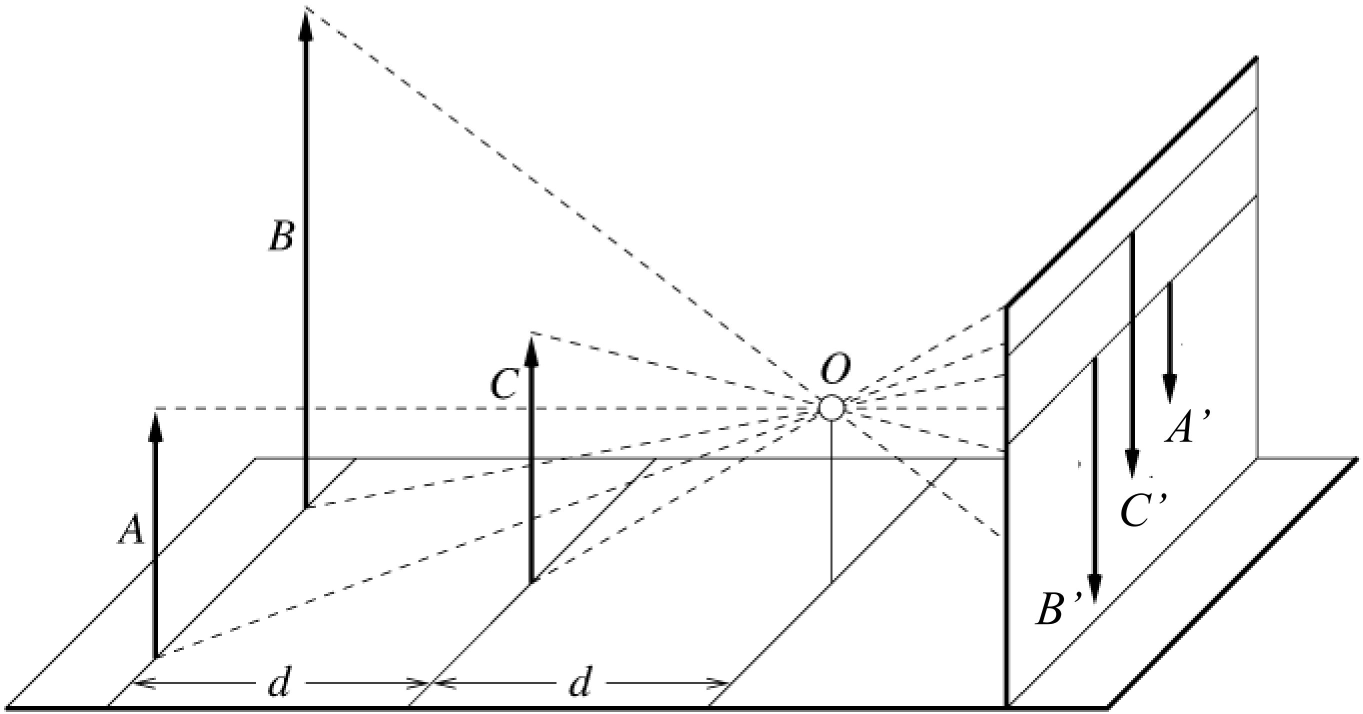
Dimensionality reduction machine



Projection can be tricky…

Projection can be tricky…

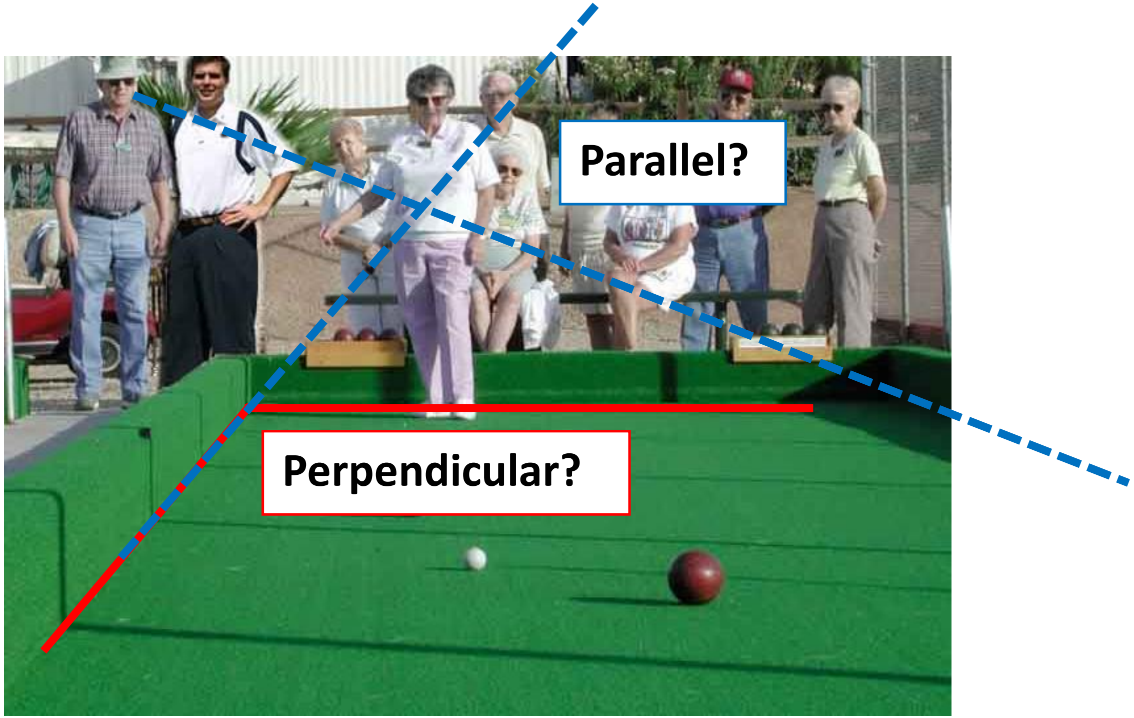
Projective geometry



Length and area are not preserved

Figure from Forsyth

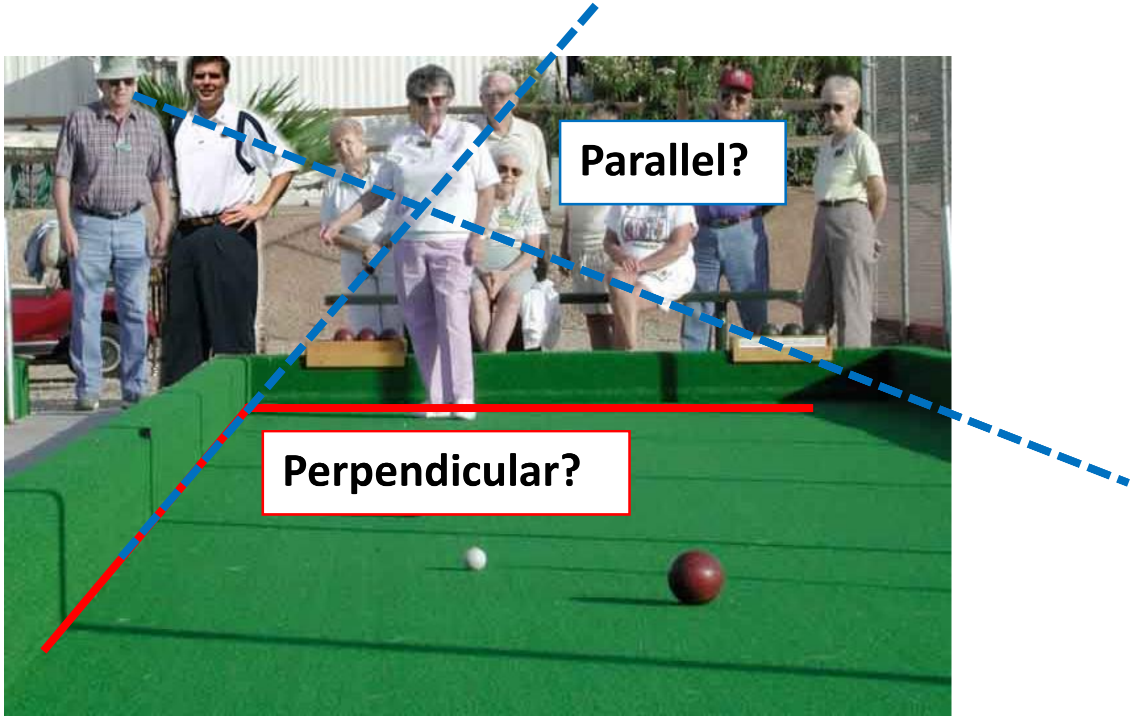
Projective geometry



What is lost?

Length and angles are not preserved

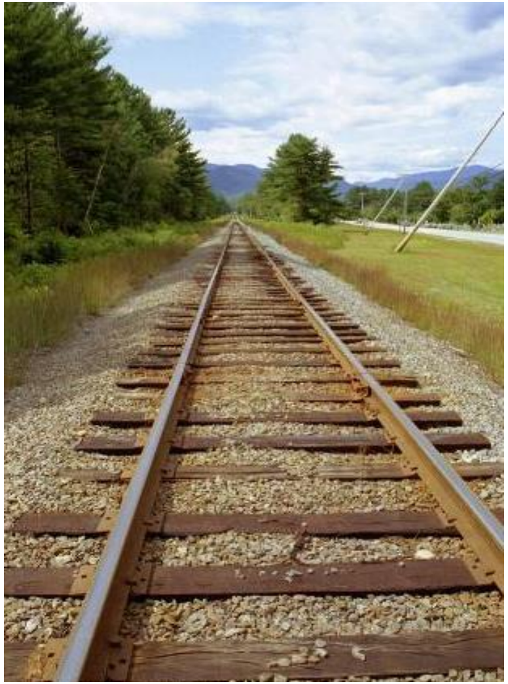
Projective geometry



What is preserved?

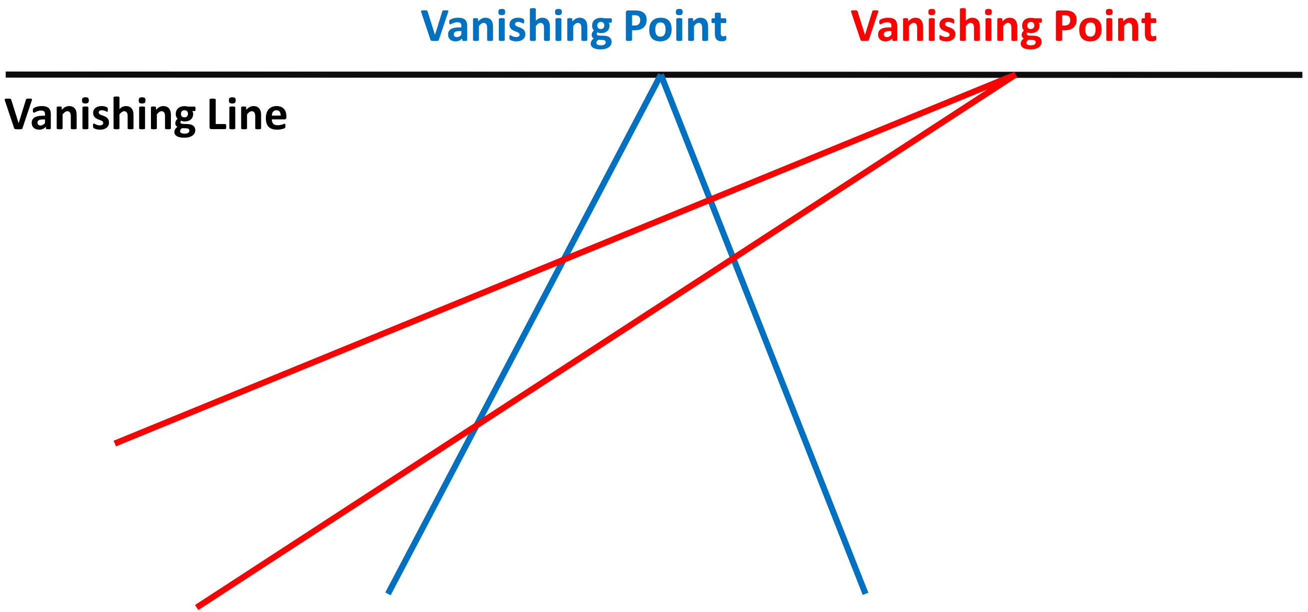
Straight lines are still straight

Vanishing points and lines



Parallel lines in the world intersect in the image at a "vanishing point"

Vanishing points and lines

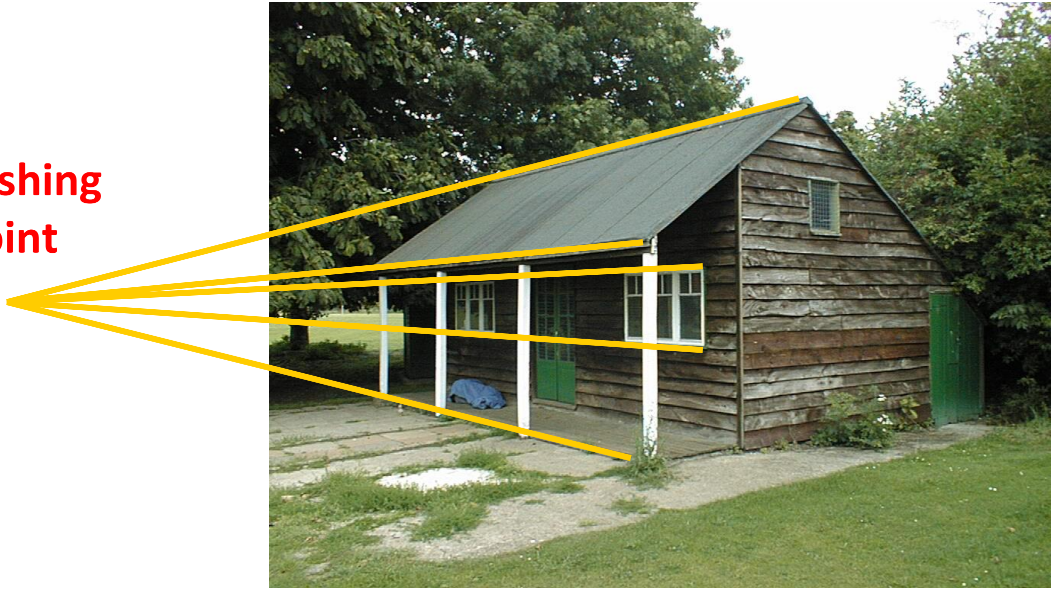


Vanishing points and lines

Slide from Efros, photo from Criminisi

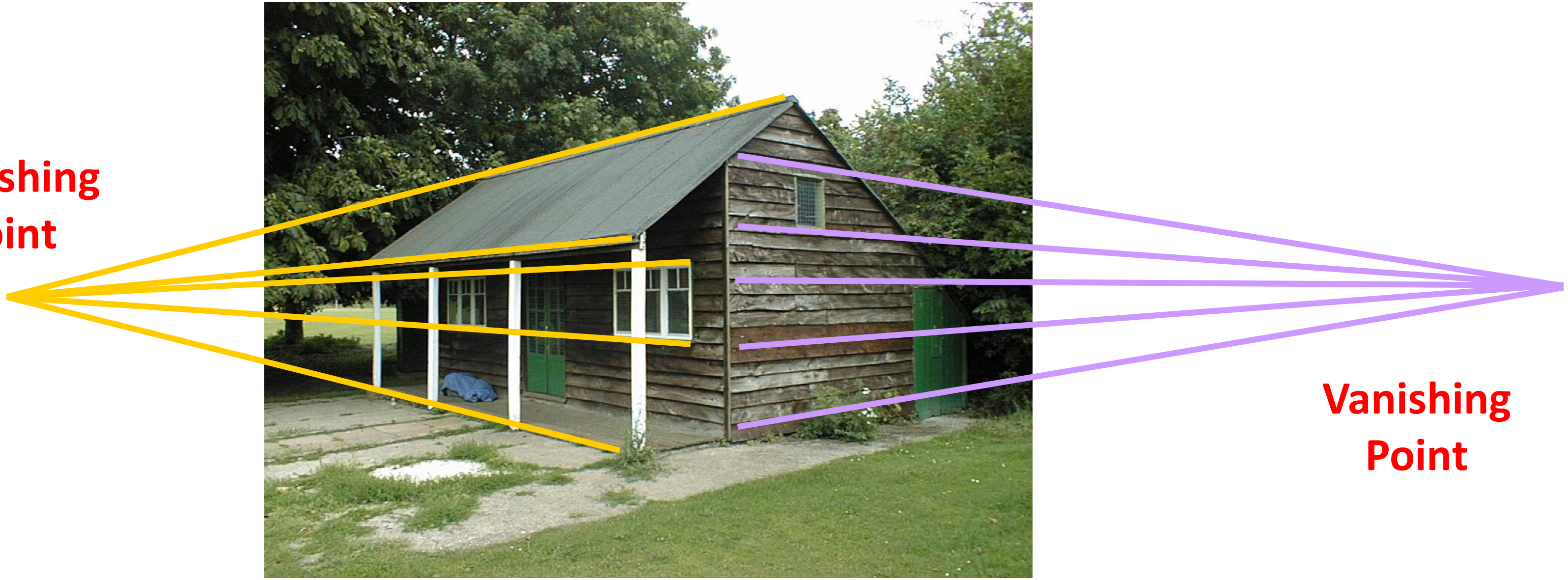
Vanishing points and lines

Vanishing Point

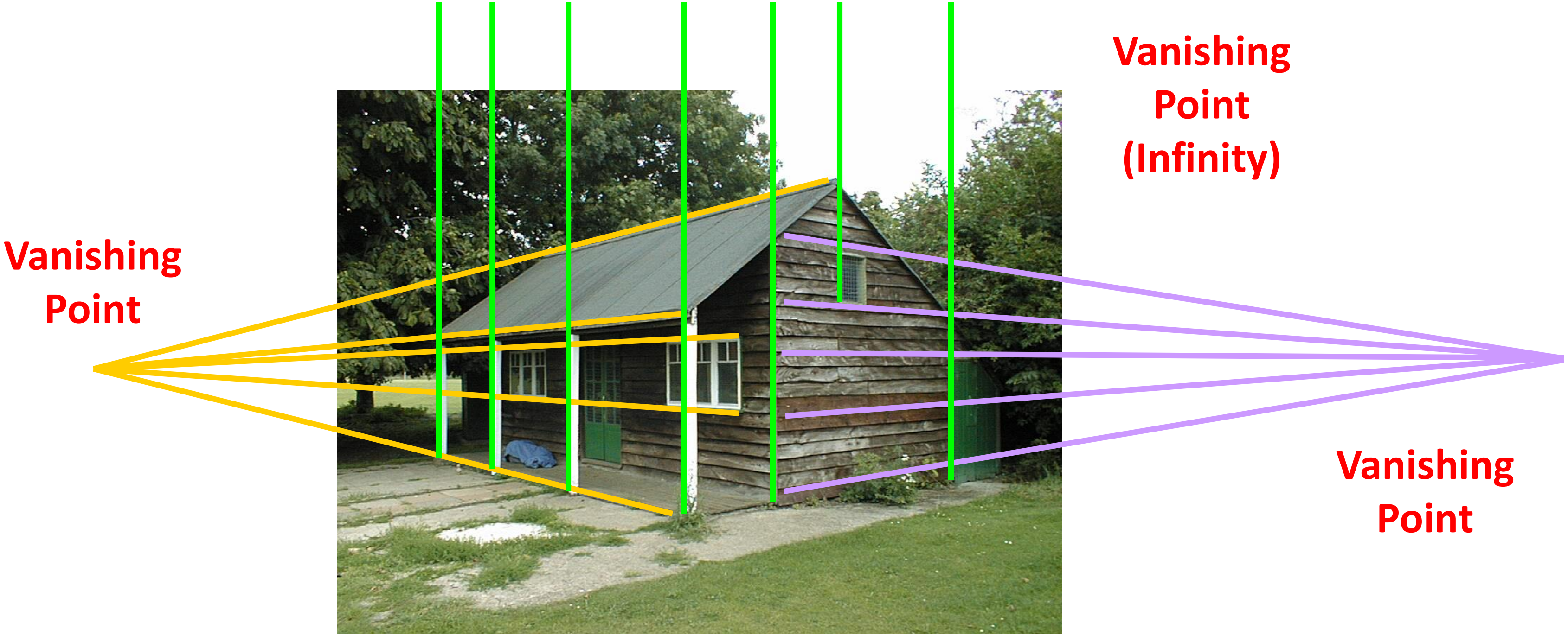


Slide from Efros, photo from Criminisi

Vanishing points and lines

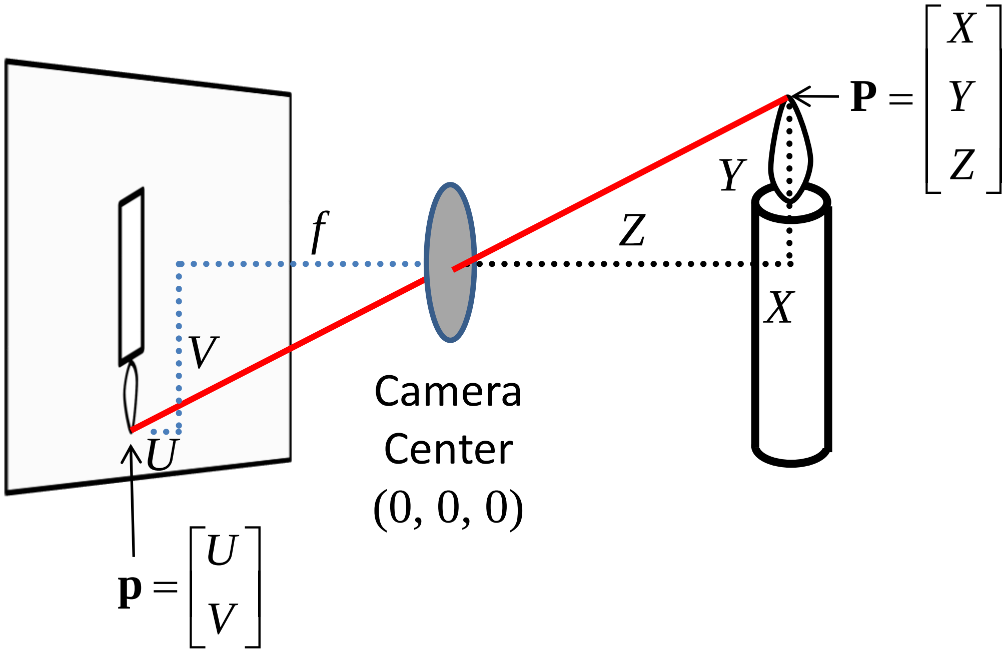


Vanishing points and lines



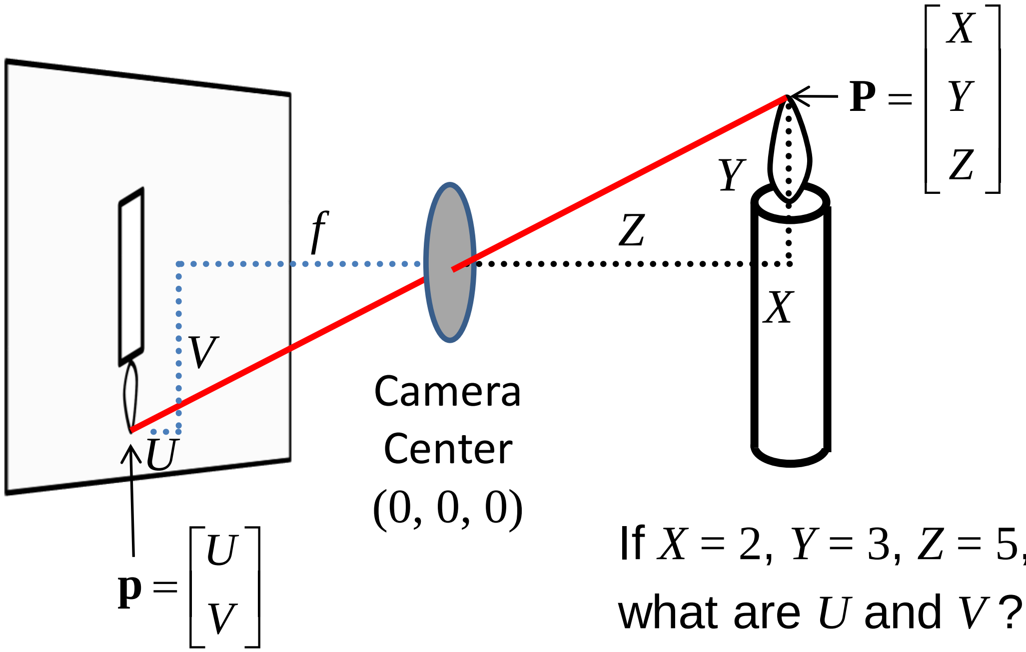
Projection maths

world coordinates => image coordinates



Projection maths

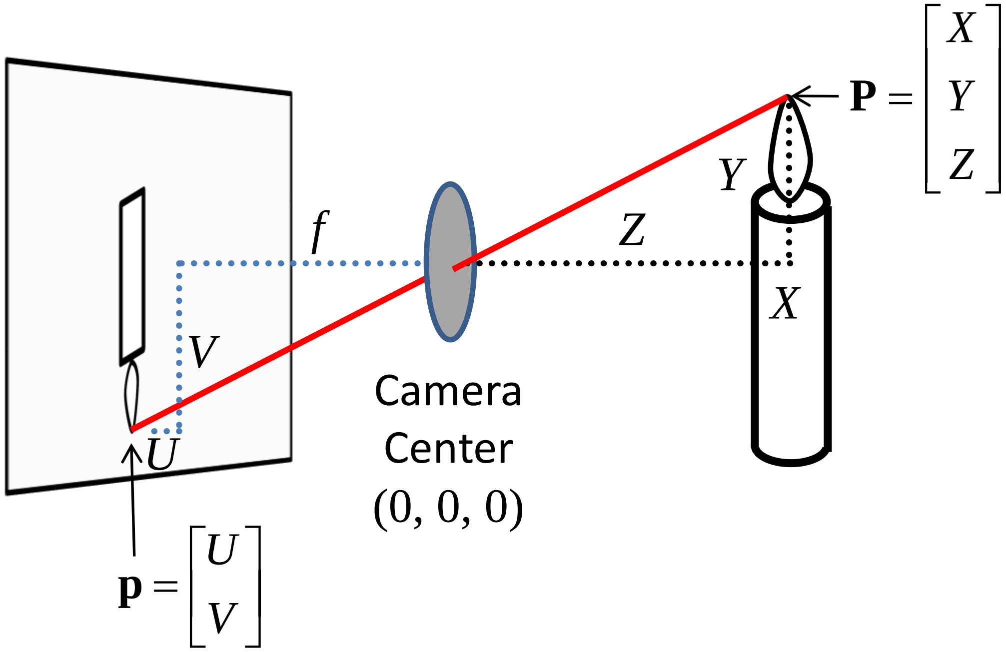
world coordinates => image coordinates



If $X = 2$, $Y = 3$, $Z = 5$, and $f = 2$,
what are U and V?

Projection maths

world coordinates => image coordinates

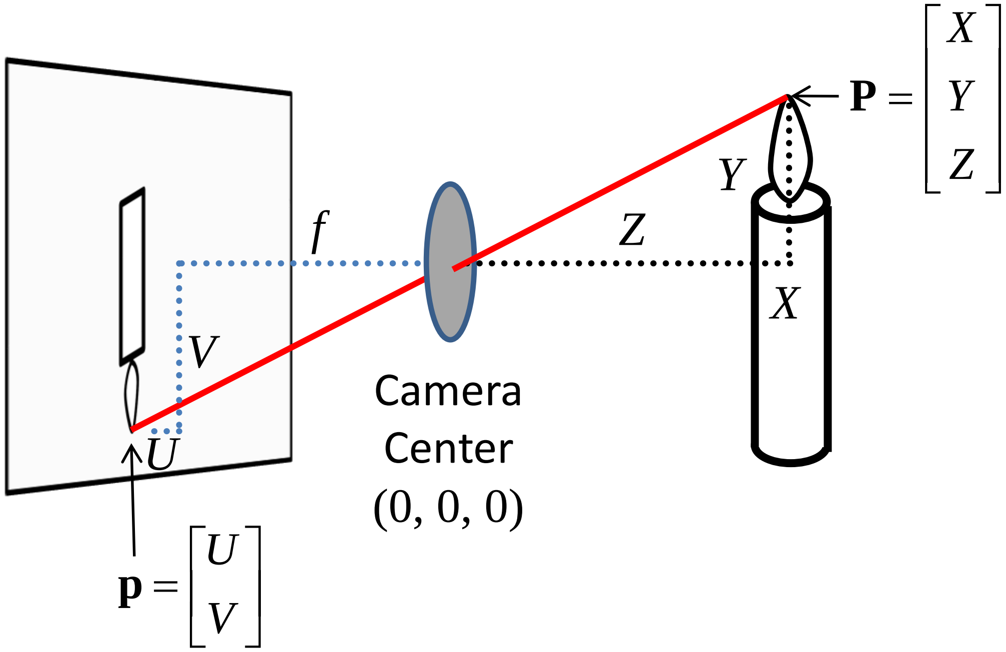


$$U = -X * \frac{f}{Z}$$

$$V = -Y * \frac{f}{Z}$$

Projection maths

world coordinates > image coordinates



$X = 2$

$Y = 3$

$Z = 5$

$f = 2$

$$U = -X * \frac{f}{Z} \qquad U = -2 * \frac{2}{5}$$

$$V = -Y * \frac{f}{Z} \qquad V = -3 * \frac{2}{5}$$

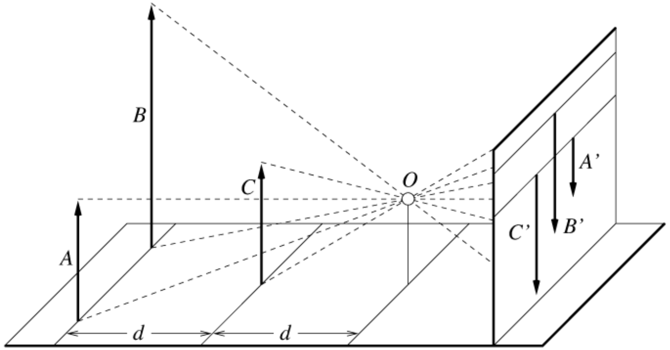
Perspective projection

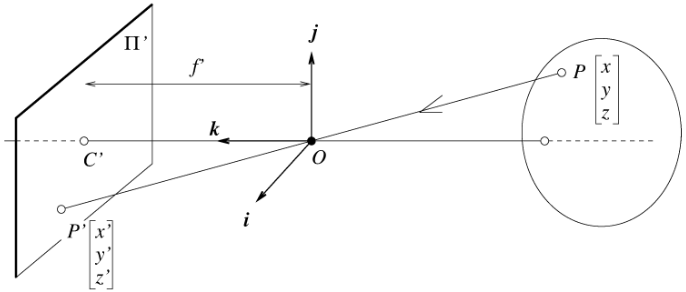
- Apparent size of object depends on its distance: far objects appear smaller
- By similar triangles

$$(x', y', z') = (f\frac{x}{z}, f\frac{y}{z}, -f)$$

- Ignore the third coordinate

$$(x', y') = (f\frac{x}{z}, f\frac{y}{z})$$

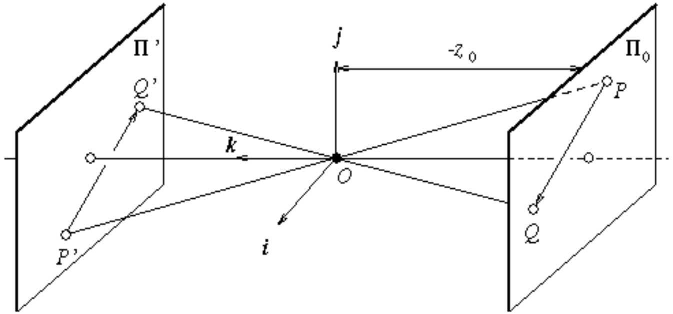




Affine projection

- Suitable when scene depth is small relative to the average distance from the camera
- Let magnification $m = -f'/z_0$ be a positive constant, treat all points in the scene as at constant distance z_0 from camera
- Leads to weak perspective projection

$$(x', y') = (-mx, -my)$$

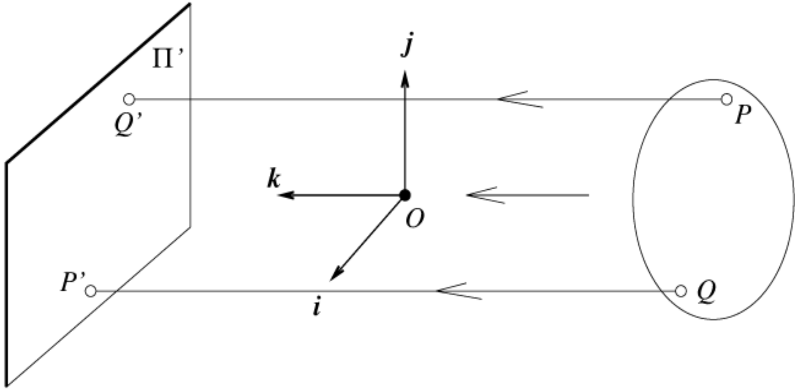


Affine projection

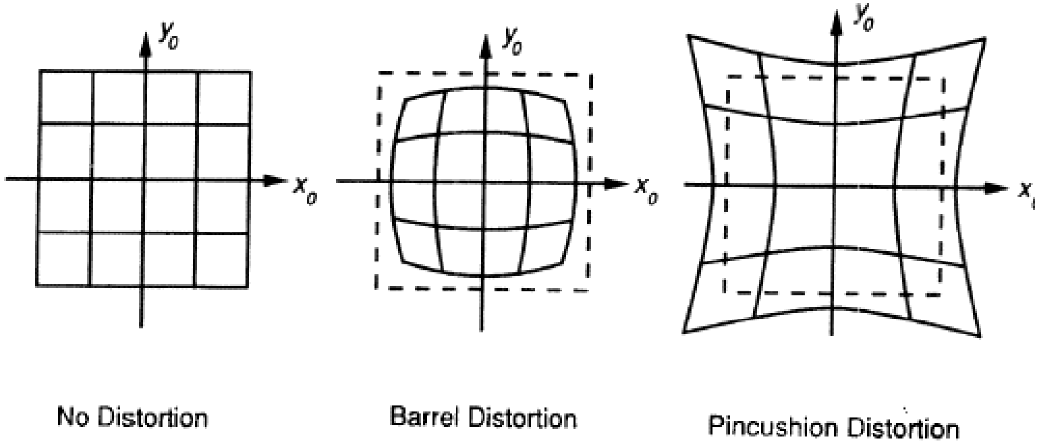
- Camera always remains at roughly constant distance from the scene

- Orthographic projection when m is normalised to -1

$$(x', y') = (x, y)$$



Beyond pinholes: radial distortions



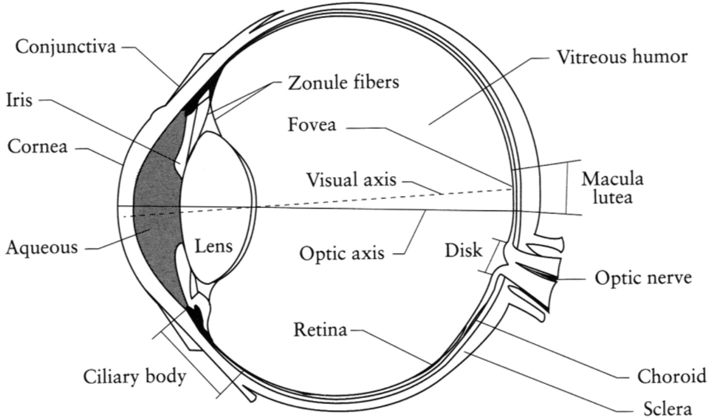
Corrected barrel distortion

 Image from Martin Habbecke

Comparing with human vision

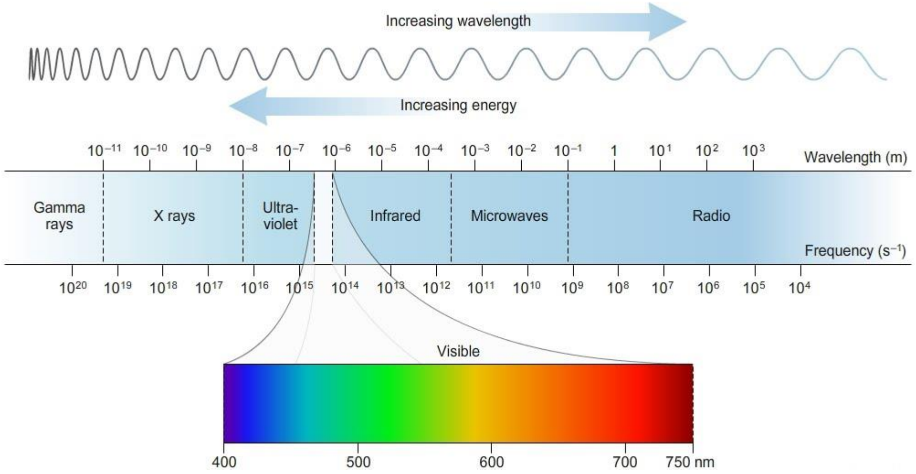
- Cameras imitate the frequency response of the human eye, so it is good to know something about it

- Computer vision probably would not get as much attention if biological vision (especially human vision) had not proven that it is possible to make important judgements from 2D images

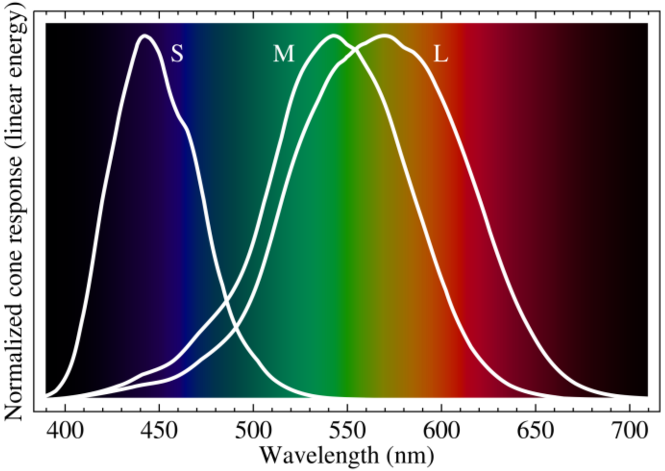


The Eye

Electromagnetic spectrum

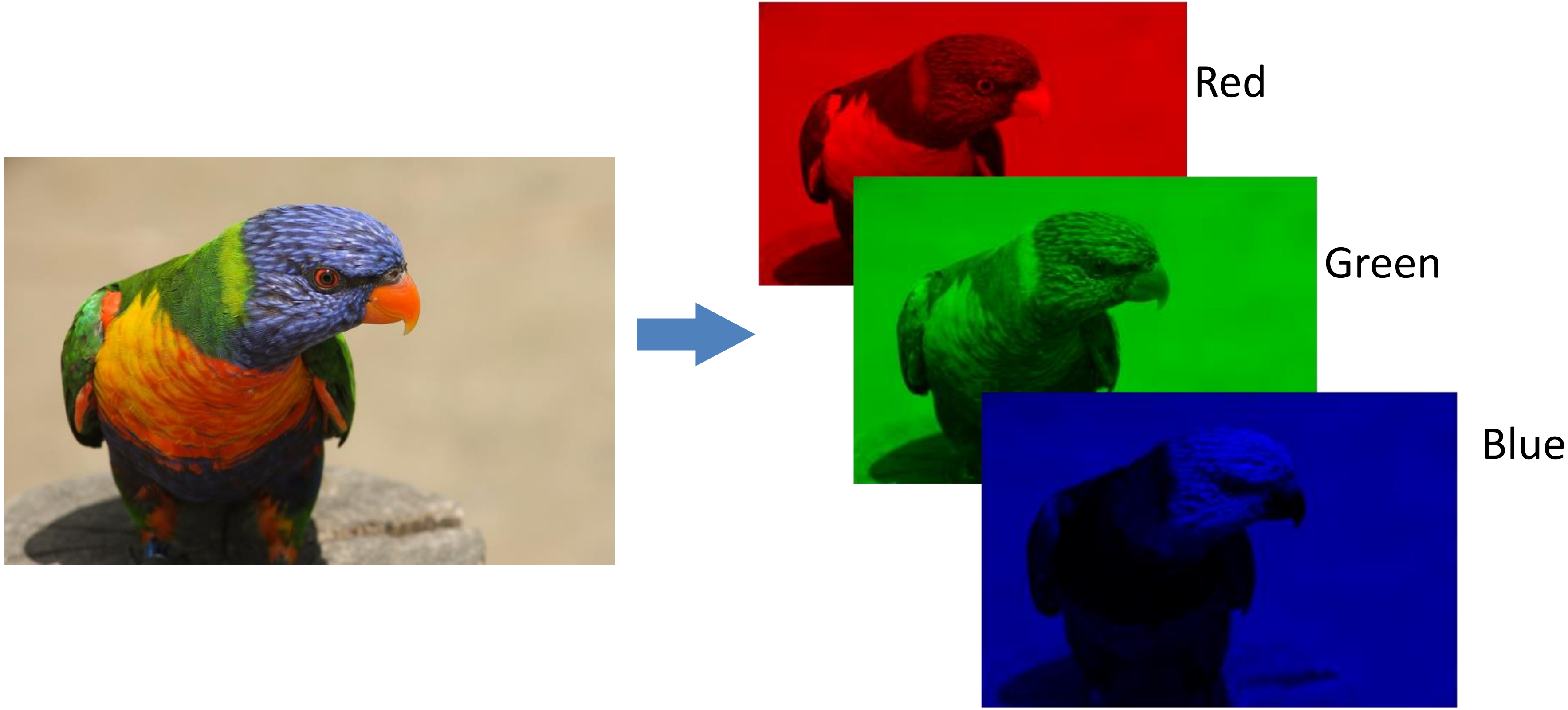


https://sites.google.com/site/chempendix/em-spectrum



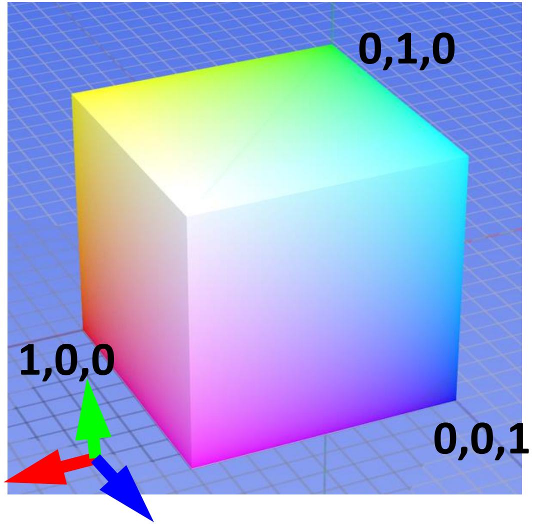
Normalized responsivity spectra of human cone cells (S, M, L types)

Colour represented by RGB images



Colour spaces: RGB

Default colour space



Drawback: strongly correlated channels

R

(G=0,B=0)

G

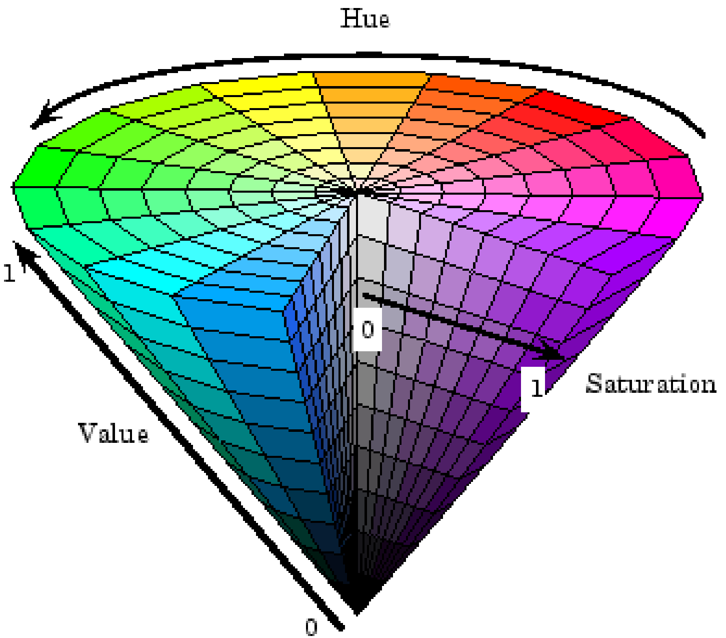
(R=0,B=0)

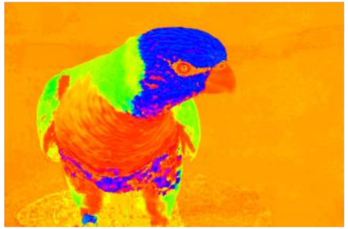
B

(R=0,G=0)

Colour spaces: HSV

Intuitive colour space





H

(S=1,V=1)

S

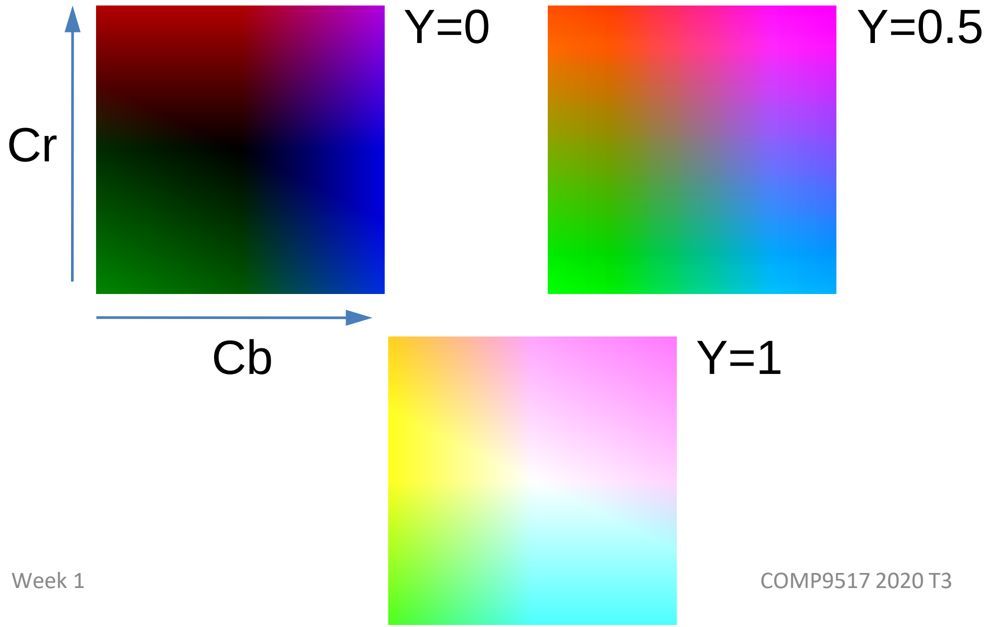
(H=1,V=1)

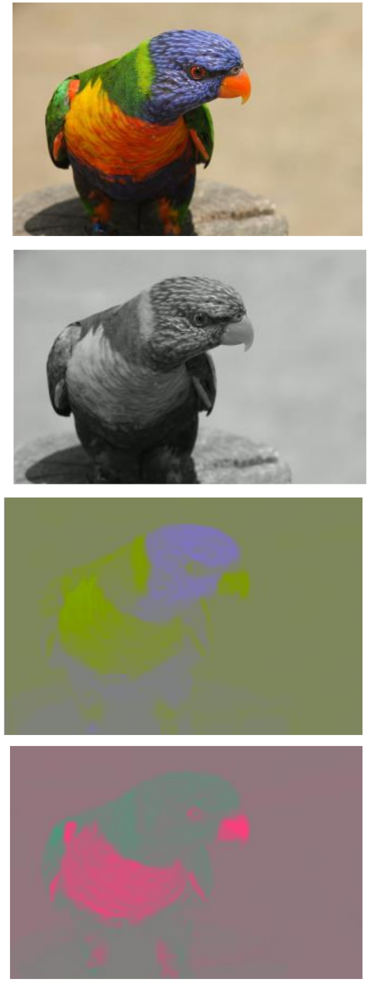
V

(H=1,S=0)

Colour spaces: YCbCr

Fast to compute, good for compression, used by TV





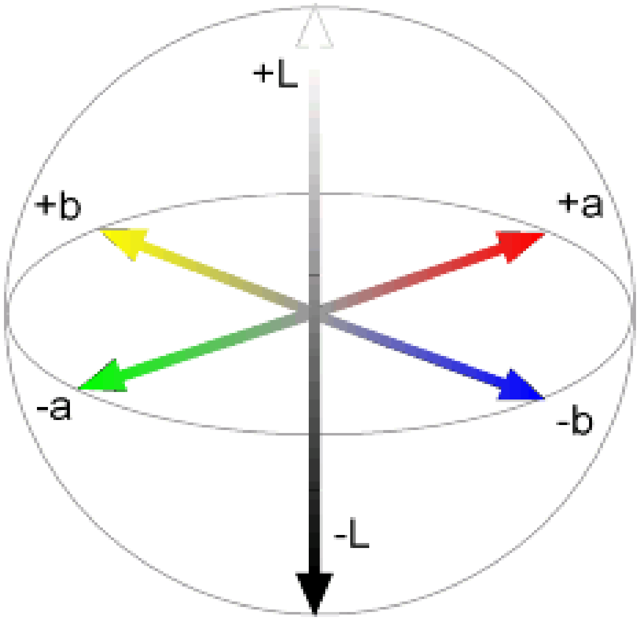
Y
(Cb=0.5,Cr=0.5)

Cb
(Y=0.5,Cr=0.5)

Cr
(Y=0.5,Cb=0.5)

Colour spaces: L*a*b*

"Perceptually uniform" colour space



L

(a=0,b=0)

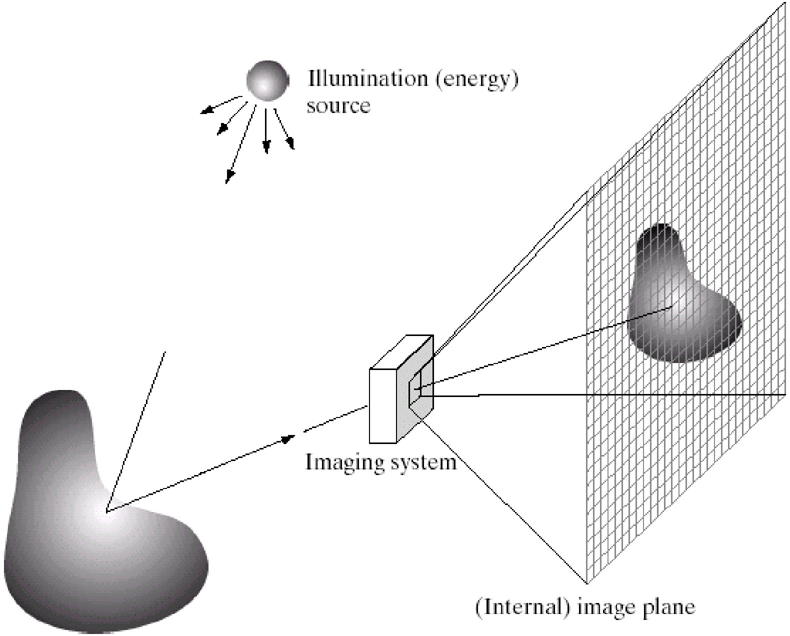
a

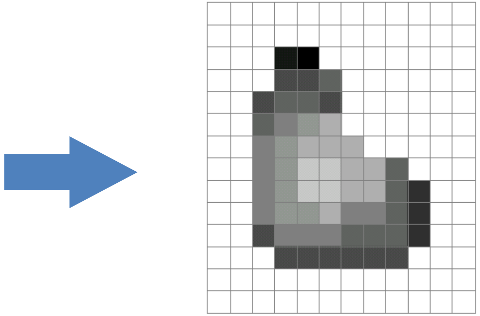
(L=65,b=0)

b

(L=65,a=0)

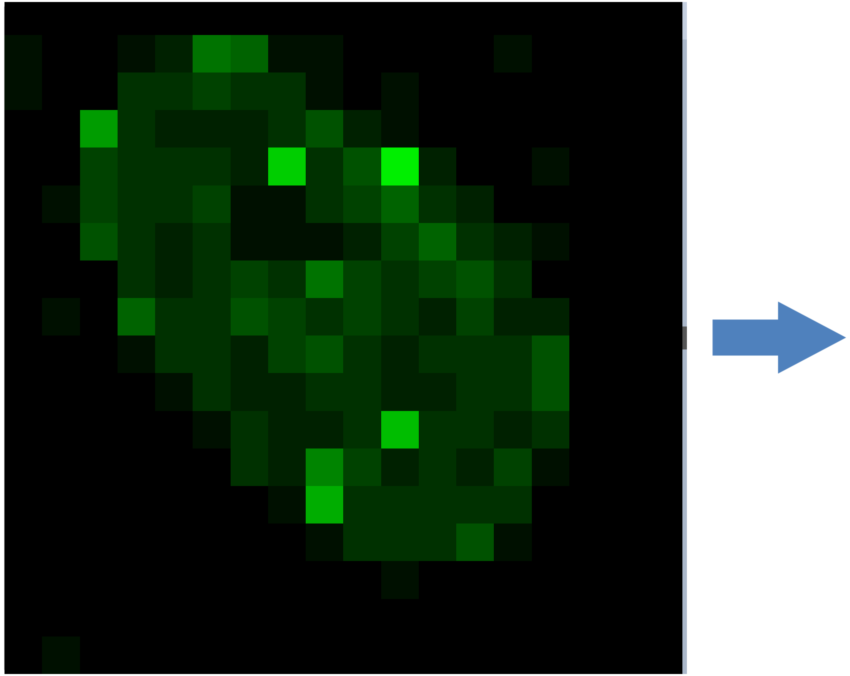
Digital image formation





Digital image

Digital image formation

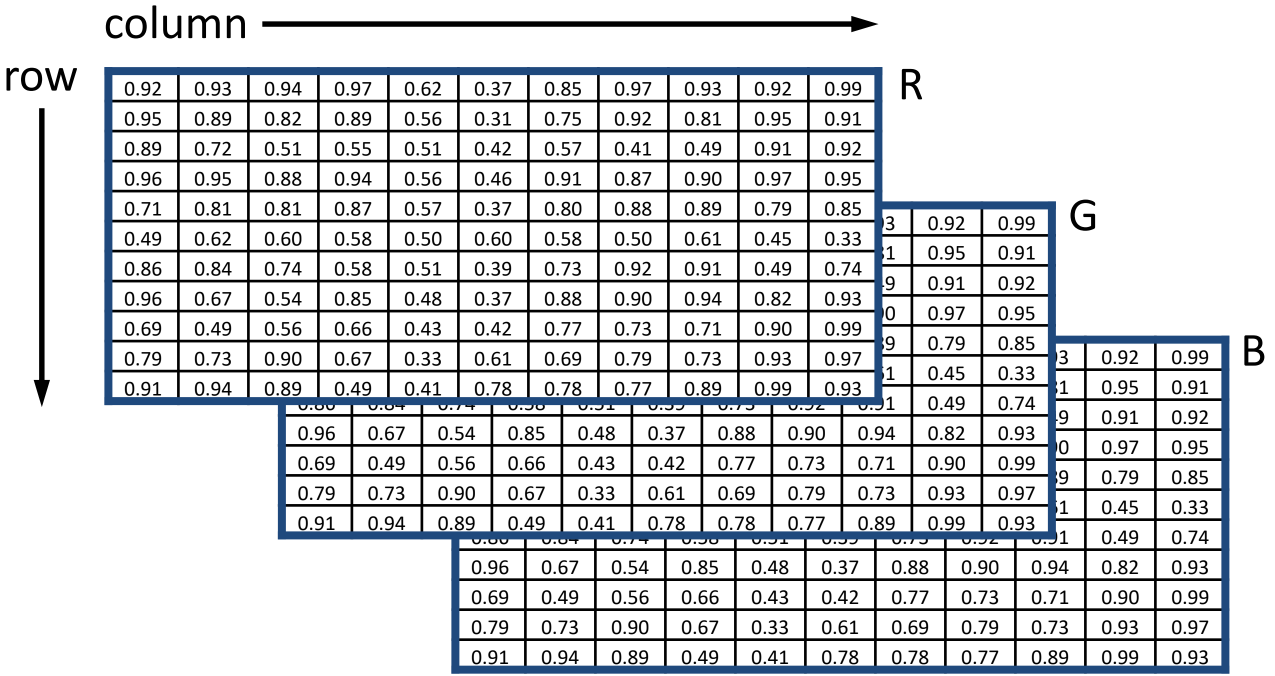


0	2	2	2	5	8	11	8	2	2	0	0	0	0	0	0	0	0
0	0	2	11	76	136	164	85	11	5	2	2	0	0	0	0	0	0
0	2	25	172	181	133	133	164	90	14	5	2	2	0	0	0	0	0
2	5	175	130	104	127	141	164	206	65	31	11	2	2	0	0	0	0
2	28	212	124	110	204	164	232	133	155	218	87	14	2	2	0	0	0
2	73	178	133	121	195	34	31	198	175	204	167	104	14	5	0	0	0
2	45	226	141	113	184	53	59	70	192	133	138	167	99	11	2	0	0
0	2	70	184	102	116	155	161	175	155	141	184	255	138	34	5	2	0
0	0	5	141	121	133	209	215	133	206	124	121	130	153	104	8	2	0
0	0	2	73	164	124	121	198	252	147	121	127	119	119	150	19	2	0
0	0	0	5	93	150	102	119	130	127	104	121	124	133	153	25	2	0
0	0	0	0	5	62	153	155	119	136	198	155	127	124	187	19	2	2
0	0	0	0	0	5	138	161	178	155	127	124	141	158	232	5	2	2
0	0	0	0	0	0	11	113	164	172	184	102	121	164	79	2	2	2
0	0	0	0	0	0	2	5	36	206	187	147	164	153	5	2	2	0
0	0	0	0	0	0	0	0	2	5	25	76	31	2	2	2	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Digitisation by spatial sampling

- **Digitisation** converts an analog image to a digital image by sampling the image space
- Sampling digitises the coordinates x and y:
  - Spatial discretisation of a picture function $F(x,y)$
  - Uses a (typically rectangular) grid of sampling points: $x = j\Delta x, y = k\Delta y \;\; | \;\; j = 1...M, k = 1...N$
  - The Δx, Δy are called the **sampling intervals**

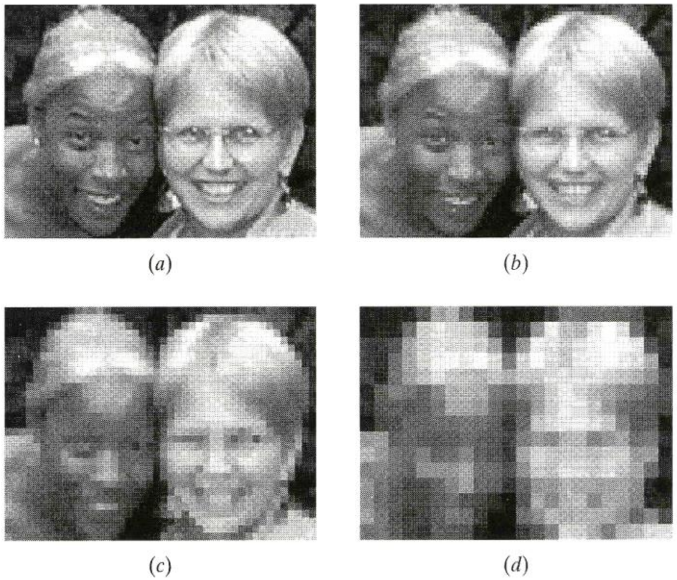
Digital colour images



0.92	0.93	0.94	0.97	0.62	0.37	0.85	0.97	0.93	0.92	0.99
0.95	0.89	0.82	0.89	0.56	0.31	0.75	0.92	0.81	0.95	0.91
0.89	0.72	0.51	0.55	0.51	0.42	0.57	0.41	0.49	0.91	0.92
0.96	0.95	0.88	0.94	0.56	0.46	0.91	0.87	0.90	0.97	0.95
0.71	0.81	0.81	0.87	0.57	0.37	0.80	0.88	0.89	0.79	0.85
0.49	0.62	0.60	0.58	0.50	0.60	0.58	0.50	0.61	0.45	0.33
0.86	0.84	0.74	0.58	0.51	0.39	0.73	0.92	0.91	0.49	0.74
0.96	0.67	0.54	0.85	0.48	0.37	0.88	0.90	0.94	0.82	0.93
0.69	0.49	0.56	0.66	0.43	0.42	0.77	0.73	0.71	0.90	0.99
0.79	0.73	0.90	0.67	0.33	0.61	0.69	0.79	0.73	0.93	0.97
0.91	0.94	0.89	0.49	0.41	0.78	0.78	0.77	0.89	0.99	0.93

Spatial resolution

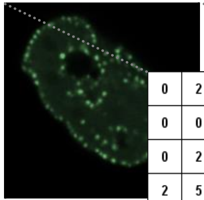
- Spatial resolution: number of pixels per unit of length
- Example: resolution decreases by one half each time (see right)
- Human faces can be recognized in 64 x 64 pixels images
- Appropriate resolution is essential:
  – Too little resolution, poor recognition
  – Too much resolution, slow and wastes memory



Quantisation

- Quantisation digitises the intensity or amplitude values $F(x,y)$
  - Called intensity or gray level quantisation
  - Gray-level resolution to be chosen
    - For example 16, 32, 64, ..., 128, 256 levels
    - Number of levels should be high enough for human perception of shading details… requires about 100 levels for a realistic image

Quantisation and bits/pixel



0	2	2	2	5	8	11	8	2	2	0	0	0	0	0	0	0	0
0	0	2	11	76	136	164	85	11	5	2	2	0	0	0	0	0	0
0	2	25	172	181	133	133	164	90	14	5	2	2	0	0	0	0	0
2	5	175	130	104	127	141	164	206	65	31	11	2	2	0	0	0	0
2	28	212	124	110	204	164	232	133	155	218	87	14	2	2	0	0	0
2	73	178	133	121	195	34	31	198	175	204	167	104	14	5	0	0	0
2	45	226	141	113	184	53	59	70	192	133	138	167	99	11	2	0	0
0	2	70	184	102	116	155	161	175	155	141	184	255	138	34	5	2	0
0	0	5	141	121	133	209	215	133	206	124	121	130	153	104	8	2	0
0	0	2	73	164	124	121	198	252	147	121	127	119	119	150	19	2	0
0	0	0	5	93	150	102	119	130	127	104	121	124	133	153	25	2	0
0	0	0	0	5	62	153	155	119	136	198	155	127	124	187	19	2	2
0	0	0	0	0	5	138	161	178	155	127	124	141	158	232	5	2	2
0	0	0	0	0	0	11	113	164	172	184	102	121	164	79	2	2	2
0	0	0	0	0	0	2	5	36	206	187	147	164	153	5	2	2	0
0	0	0	0	0	0	0	0	2	5	25	76	31	2	2	2	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Pixel (<u>pict</u>ure <u>el</u>ement)

Levels per pixel:

8 bits = 2^8 = 256

12 bits = 2^{12} = 4,096

16 bits = 2^{16} = 65,536

24 bits = 2^{24} = 16,777,216

Further reading

- Chapter 2 of Szeliski
- Chapter 2 of Shapiro and Stockman

Acknowledgements

- Several slides from Derek Hoiem, Alexei Efros, Steve Seitz, David Forsyth and Erik Meijering
- Image sources credited where possible
- Some material, including images and tables, were drawn from the referenced textbooks and associated online resources