Introduction to Computer Vision and Machine Learning

VISUM'19

Porto, PORTUGAL

Jose Costa Pereira

Research Scientist @Huawei R&D, London, UK

INESCTEC, Porto, Portugal

Thanks to Nuno Vasconcelos for the slide contents

Agenda

- Computer Vision
  - Cameras
  - Radiometry and light sources
  - Color
  - Filtering
  - Smoothing and noise
  - Edges
- Machine Learning
  - Bayes decision rule
  - Maximum Likelihood
  - Least Squares, regression
  - Clustering, k-means, EM
  - Principal component analysis
  - Support Vector Machines

Why Computer Vision?

- Images are important
  - web, networks, social media, images are available everywhere
  - video is a driving force for many innovations (e.g. networking, cell-phones)
- Vision is a generalization of image processing
  - Image Processing aims to make images look better (compressed, restored, etc.), vision aims at understanding them.
  - if you can understand you can make them look better (not clear how to do that otherwise…)
- Vision gives you a broader exposure
  - lies at the intersection of deterministic (filtering, convolutions, Fourier, etc) and statistical signal processing (expectations, covariances, classification error)

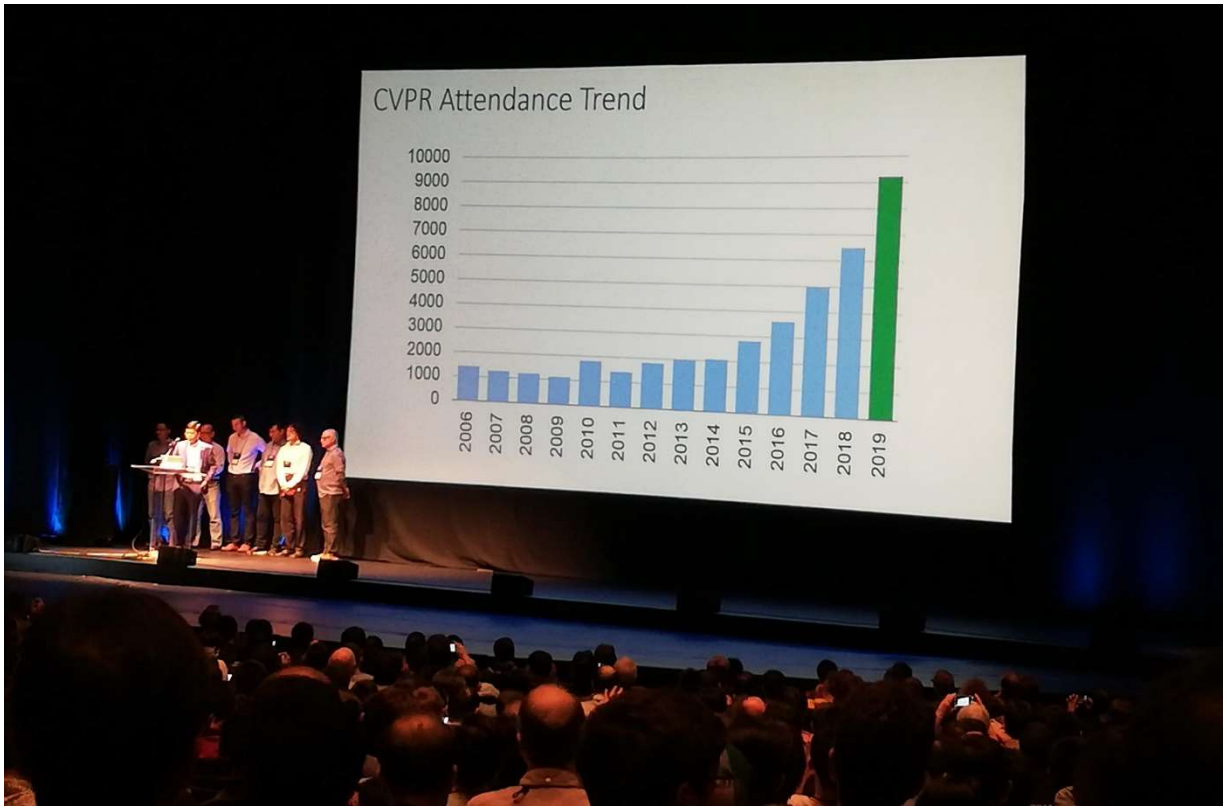
Why Computer Vision? (contd.)

- Vision is fundamentally harder
  - in the absence of biological vision systems, we would have given up a long time ago
  - more science to be done
- Connections to cognitive science
  - interest/familiarity with literature from other areas
  - broader exposure
- Vision has a broader range of applications
  - desired skill-set in the industry
    - e.g. Self-driving vehicles, fashion industry, IoT, computational photography, etc.
  - you will be more likely to get a job

Why Computer Vision? (contd.)

- and if nothing else… because it's popular!

CVPR '19



Applications

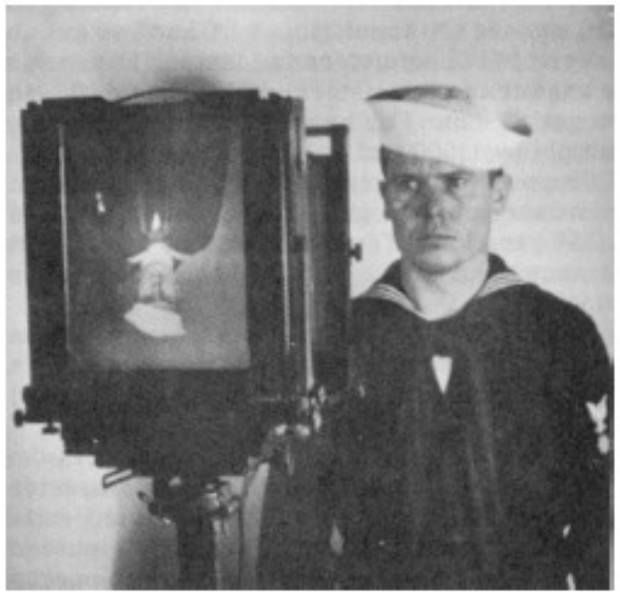
- Scene understanding
- Object recognition
- Understanding actions
- Surveillance
- Medical imaging
- Computational photography
- Autonomous driving

Topics

- Image formation:
  - how do images get captured?
- Low-level vision:
  - basically filtering
  - much information can extracted by filtering images in different ways.
- Mid-level vision:
  - what are the components of an image? (segmentation)
  - how can we fit models to our images/video?
  - what is the scene motion and how do we measure it?
- High-level vision:
  - a lot of statistics
  - how do we find people, skin, various objects
  - how do we classify scenes?

Image formation

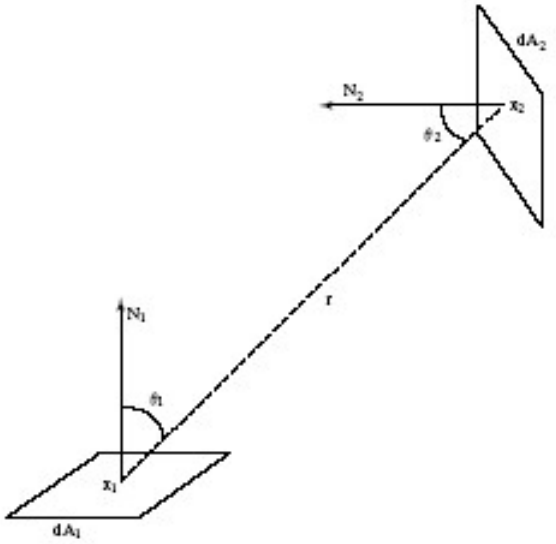
- What equations govern the projection from the 3D world to the 2D image plane?

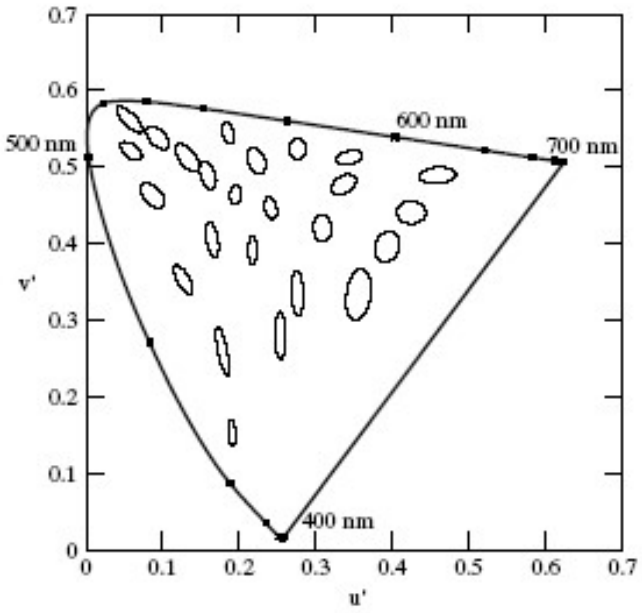


- What types of light sources are there? Does that matter?

Image formation

- How do we measure the amount of light emitted by each object?

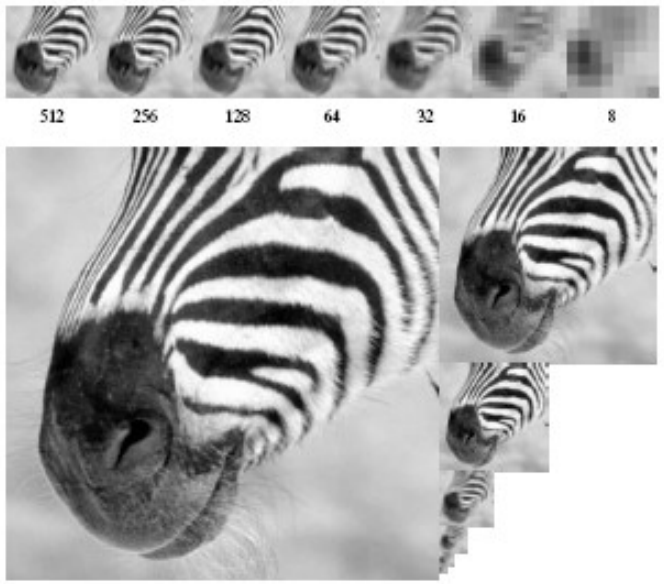


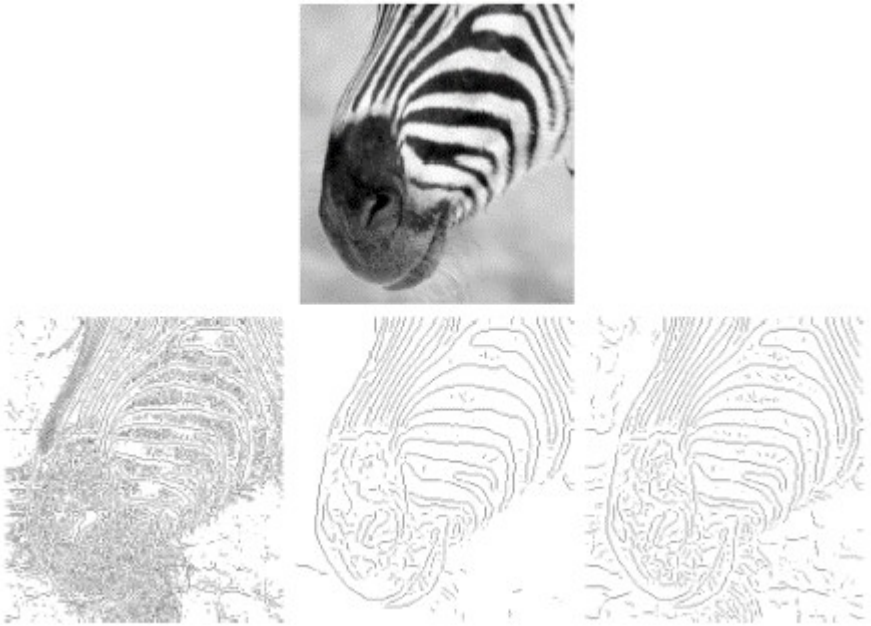


- How do we measure color? What is a color space?

Low-level vision

- The importance of scale, pyramids and multi-resolution

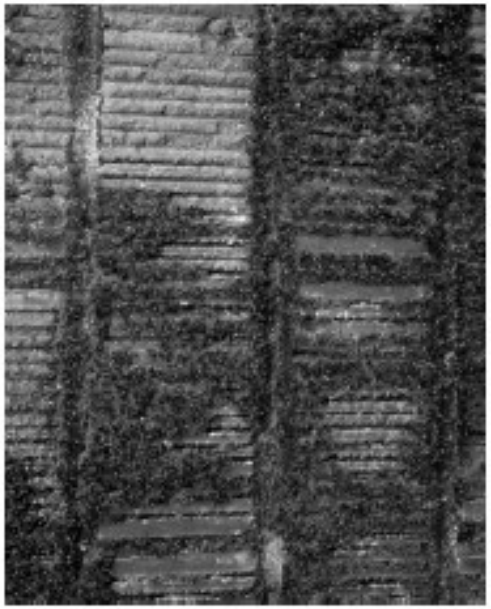




- Filtering to get derivatives, edges, corners, etc.

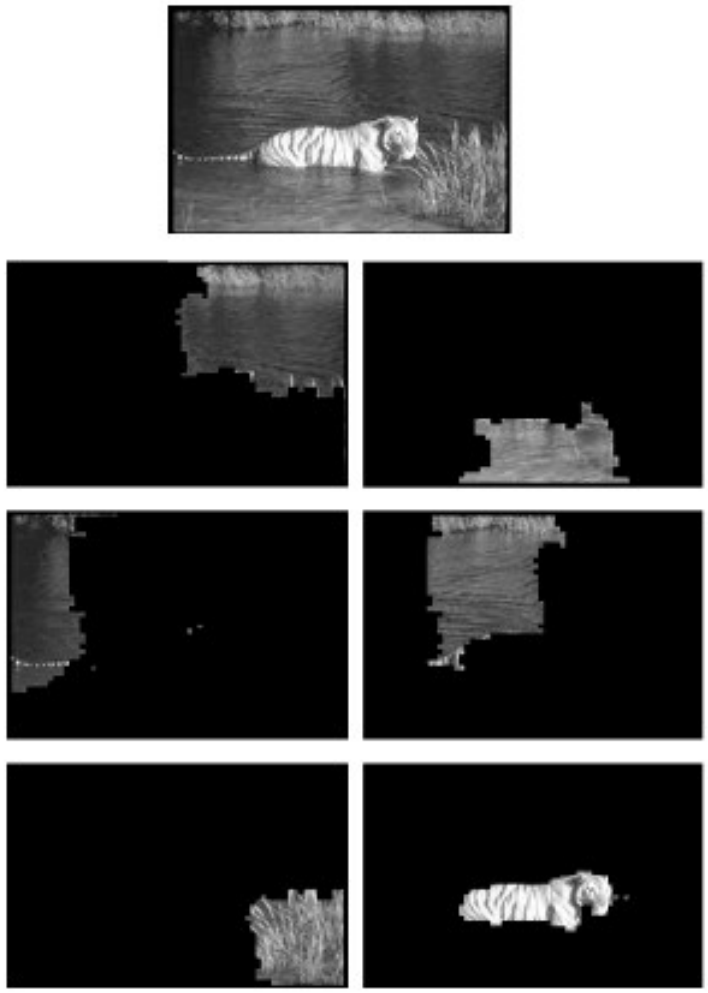
Low-level vision (contd)

- What is texture and how do we characterize, classify or segment it?



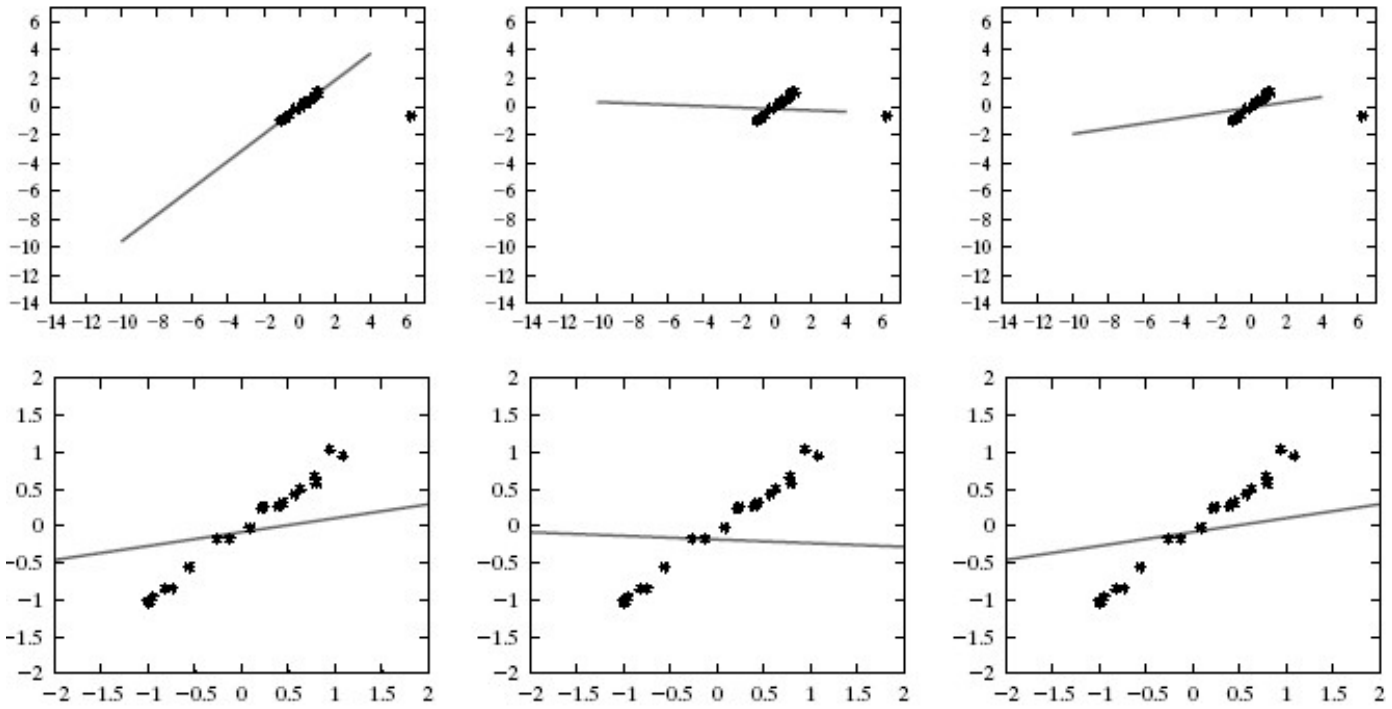
Mid-level vision

- From localized to global processing
- Tasks that are essential to achieve vision goals, but not goals by themselves
- We will study problems like grouping and segmentation
- Most of these are incredibly hard



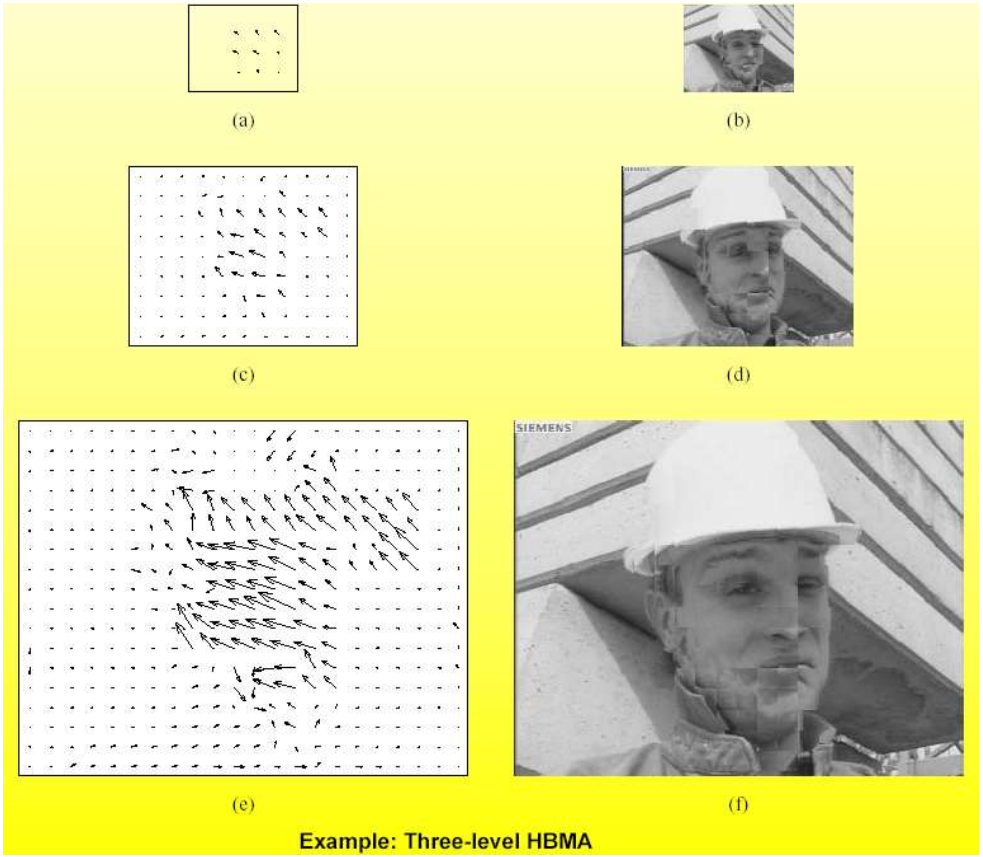
Mid-level vision (contd)

- One common solution is to fit a model to the images
- How does one minimize the error of the fit?



Mid-level vision (contd)

- Motion is an important cue for many tasks:
  - surveilance
  - compression
  - action recognition
  - registration
  - segmentation
- Various types:
  - global (camera)
  - local (clouds, leaves, etc)
  - object (people, cars)
  - rigid/non-rigid (cars vs people)
- Many algorithms



High-level vision

- Solves problems that are end-goals of vision.
- Involves lots of classification /categorization problems
- Mostly based on statistical modeling
- E.g. how do we find if there is human skin in the image?

High-level vision (contd)

- What about peoples faces? Or cars?

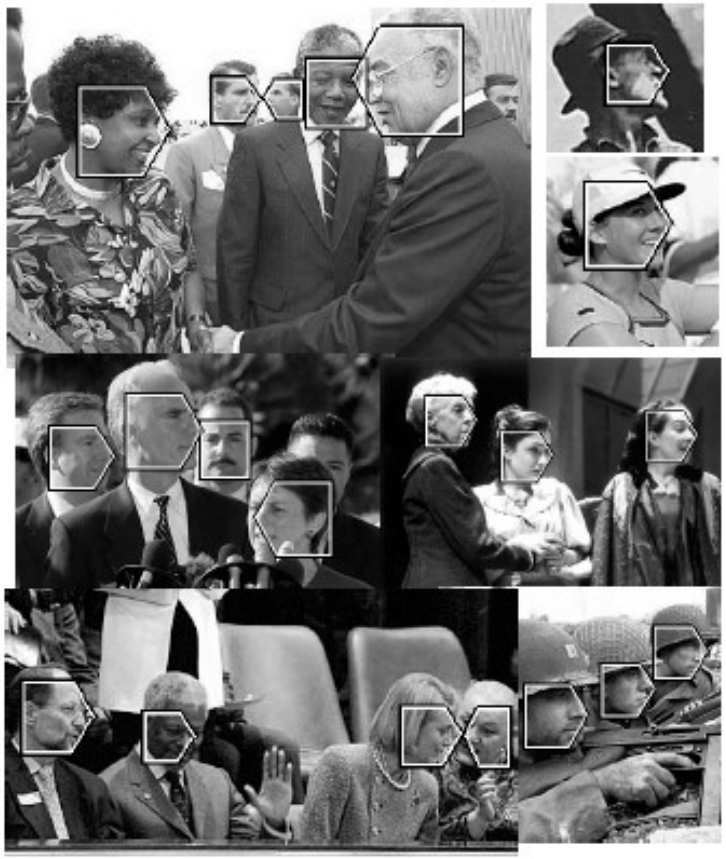
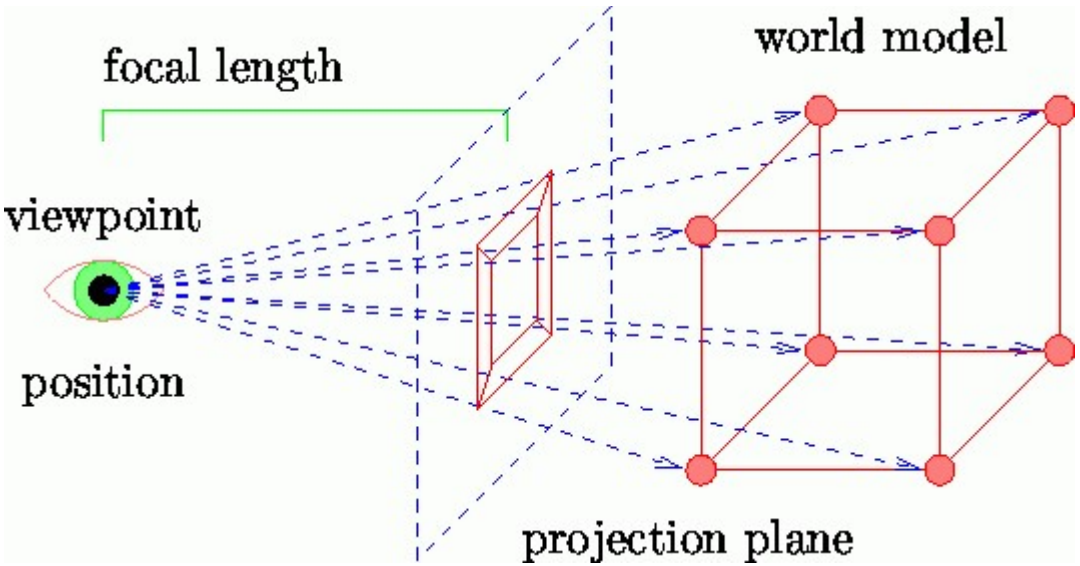


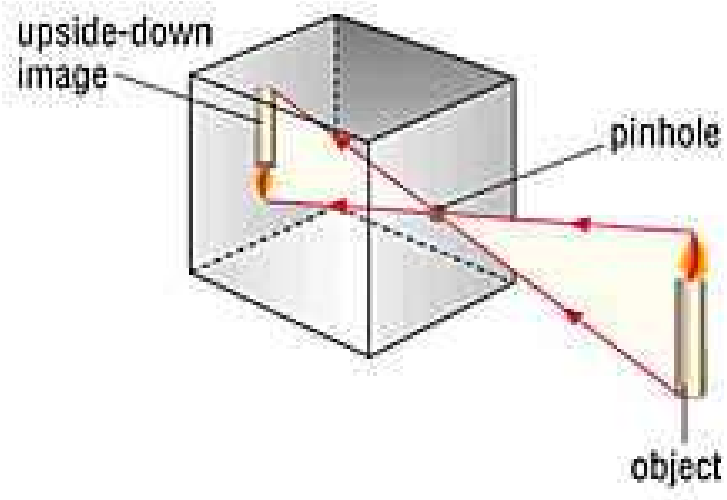
Image formation

- Image understanding starts with understanding of image formation:
  - projection of a scene from 3D world into image on 2D plane



Pinhole camera

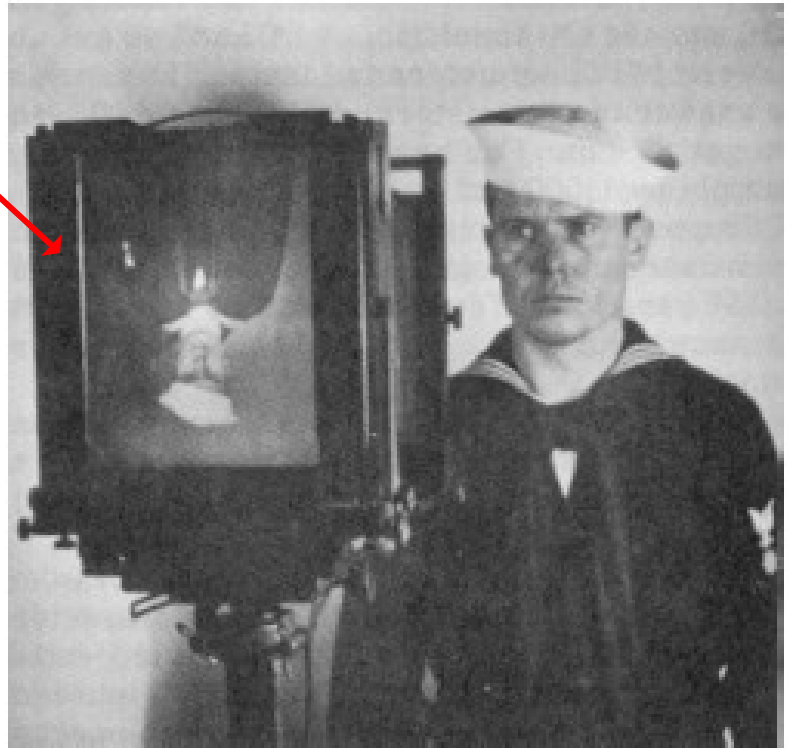
- we assume that a camera is
  - a black box
  - with an infinitesimally small hole on one face
  - the hole is so small that only one ray of light passes through it and hits the other side



real pinhole camera made by Kodak for schools, circa 1930

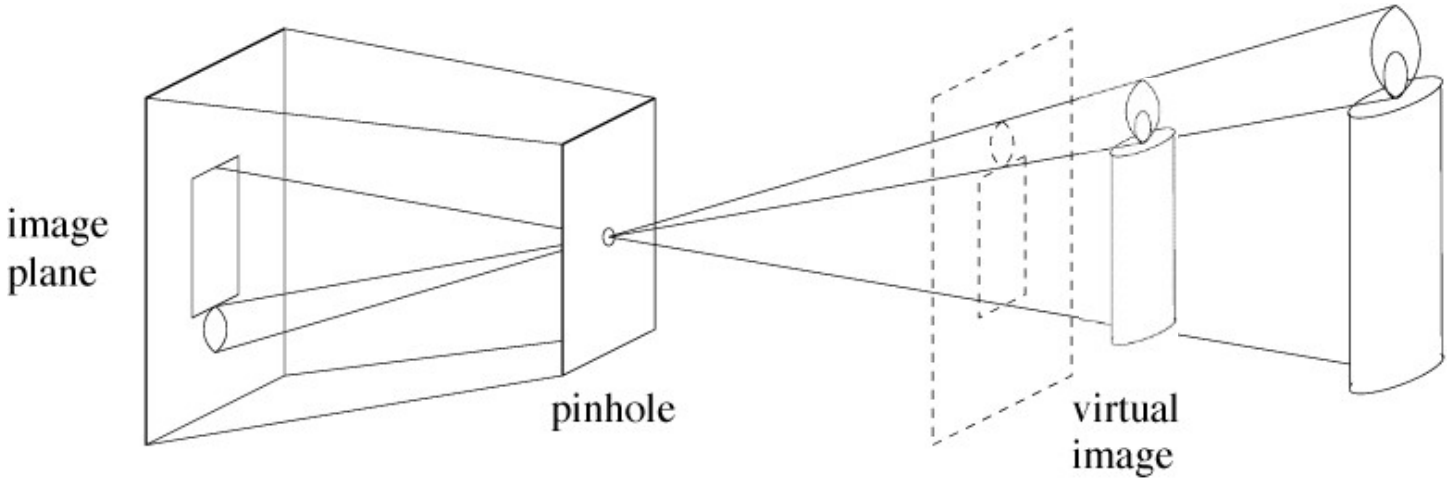
Pinhole camera (contd.)

- by placing photo-sensitive material in the back wall you will get an upside-down replica of the scene
  - this is the image plane
- to avoid the mathematical inconvenience of this inversion:
  - we consider a plane outside of the camera
  - this is called the virtual image plane



Pinhole camera (contd.)

- The virtual image plane
  - it is an abstraction
  - exactly the same as the image plane, with the exception that there is no inversion

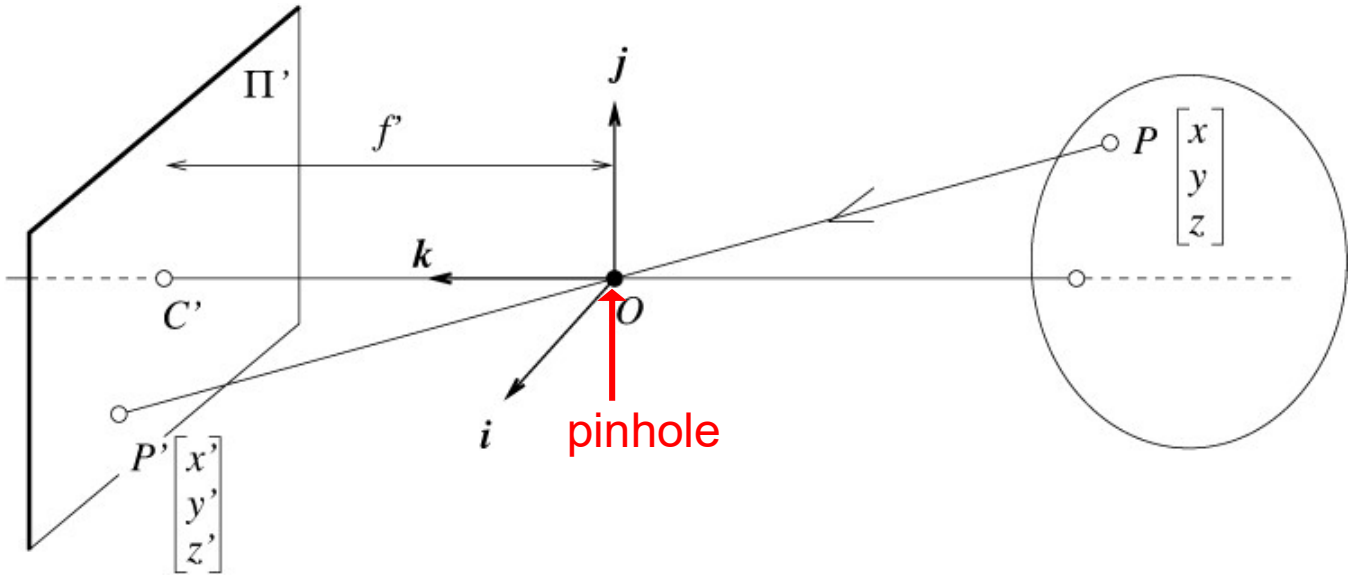


- An important property:
  - objects that are far away become smaller in the image plane

- We suspect that distance (d) to the camera plays an important role in perspective projection
  - in particular we would expect image size be inversely proportional to distance (i.e. 1/d)

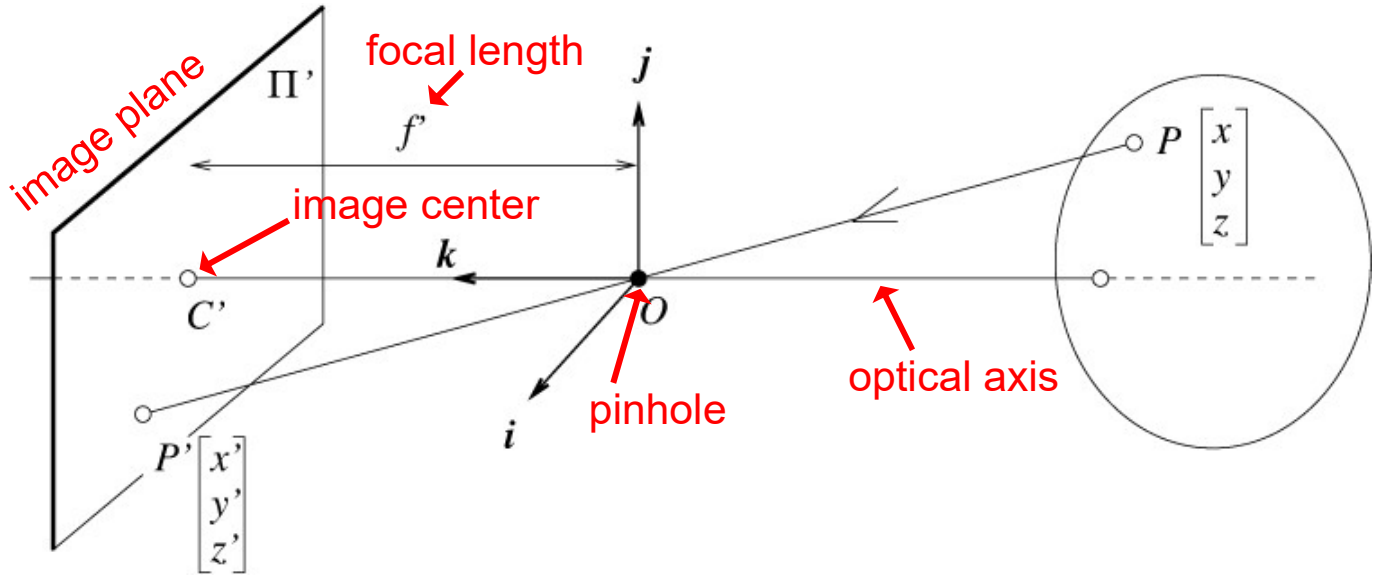
Coordinates

- to relate world point *P* to image point *P'*
  - we need a coordinate system
  - the 1/d dependence suggests using pinhole as origin
  - we also make two coordinate axes ***(i,j)*** a basis of the image plane and the third axis ***(k)*** orthogonal to it (measures depth)



Pinhole camera

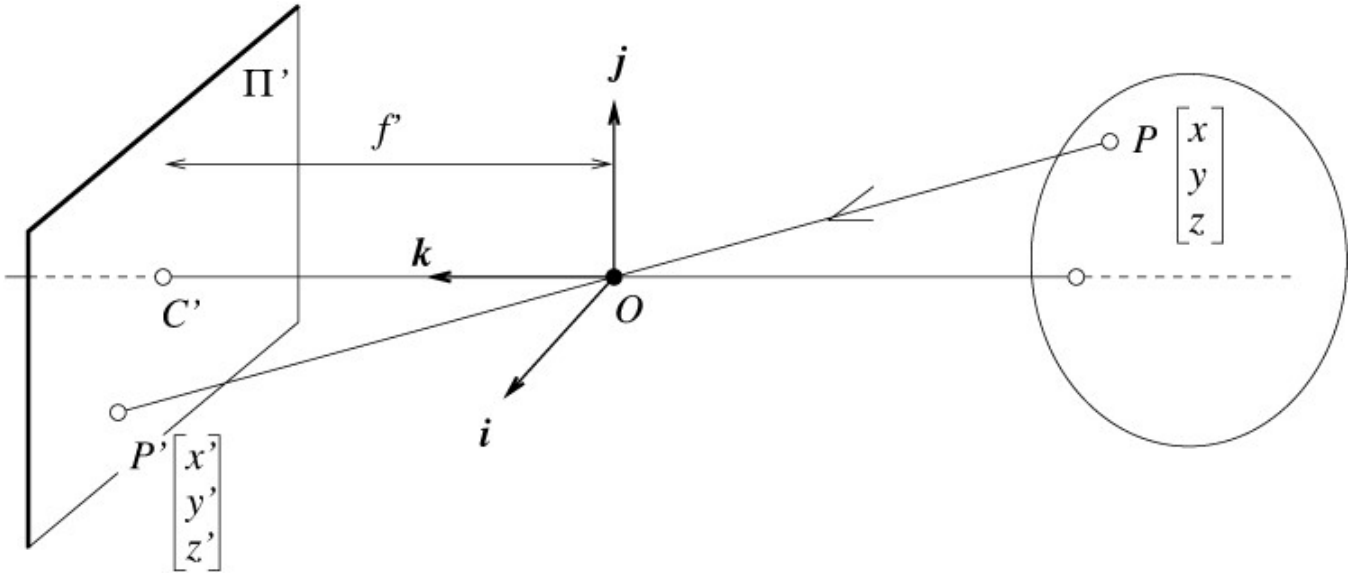
- definitions:
  - line perpendicular to image plane, through pinhole, is the optical axis
  - point where optical axis intersects image plane is the image center
  - distance f between image plane and pinhole is the focal length



Projection equations

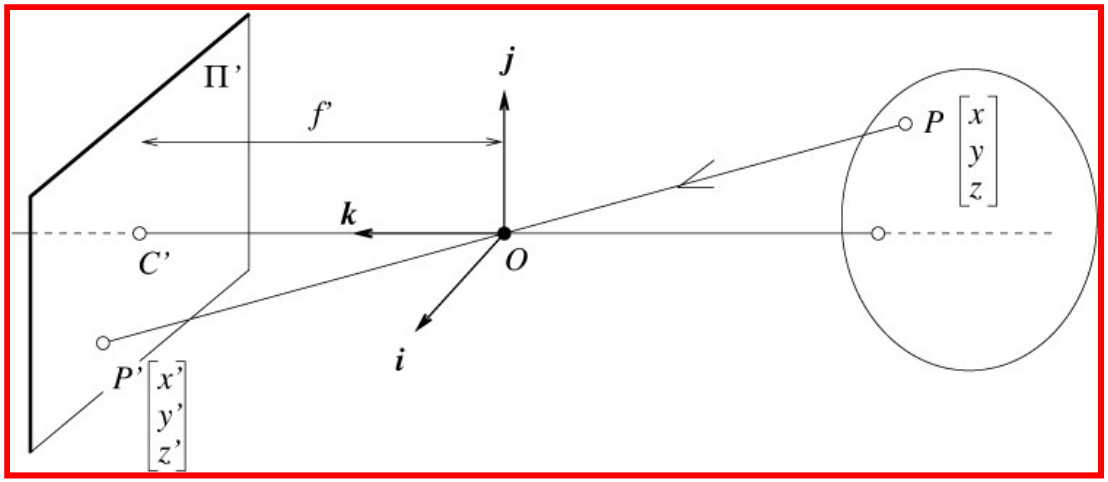
- note that
  - P, O, P' are on the same line
  - this implies that there is a λ such that $OP' = \lambda\, OP$, and

$$\begin{cases} x' = \lambda x \\ y' = \lambda y \\ f = \lambda z \end{cases} \Rightarrow \lambda = \frac{f}{z} \Rightarrow \begin{pmatrix} x' \\ y' \end{pmatrix} = f \begin{pmatrix} x/z \\ y/z \end{pmatrix}$$



Perspective projection

- this is the basic equation of perspective projection

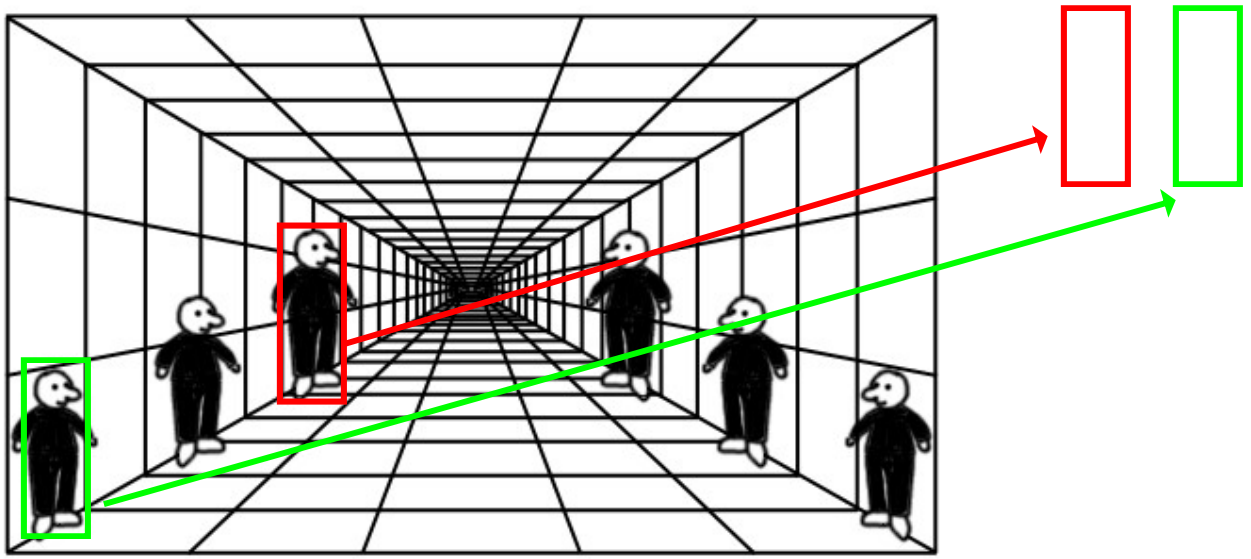


$$\begin{pmatrix} x' \\ y' \end{pmatrix} = f \begin{pmatrix} x/z \\ y/z \end{pmatrix}$$

- note that
  - there is indeed an inverse dependence on the depth Z
  - far objects become small

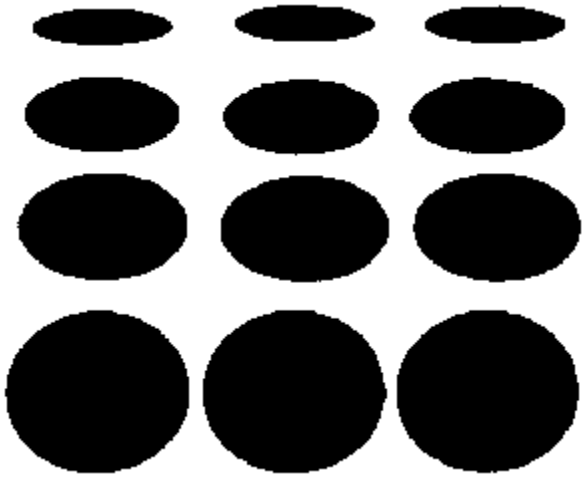
Perspective projection

- this is a very powerful cue for scene understanding
  - note that the visual system infers all sorts of properties from perspective cues
  - e.g. size

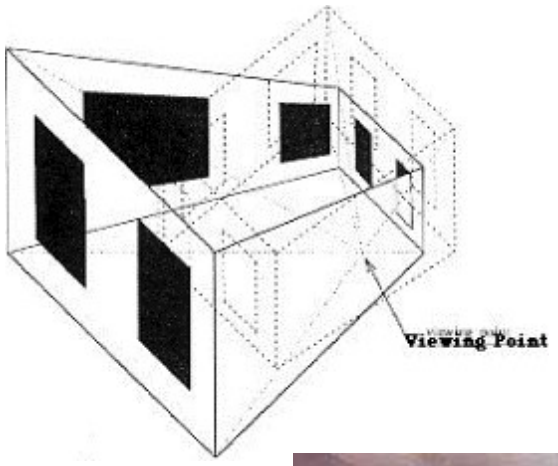


Perspective projection

- or shape



or proximity



Perspective projection

- is conceptually very simple

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = f \begin{pmatrix} x/z \\ y/z \end{pmatrix}$$

  - but is highly non-linear and usually hard to work with
  - e.g. assume you have a big plane on the scene, e.g. a wall

$$z = ax + by$$

  - then

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = f \begin{pmatrix} x/(ax+by) \\ y/(ax+by) \end{pmatrix}$$

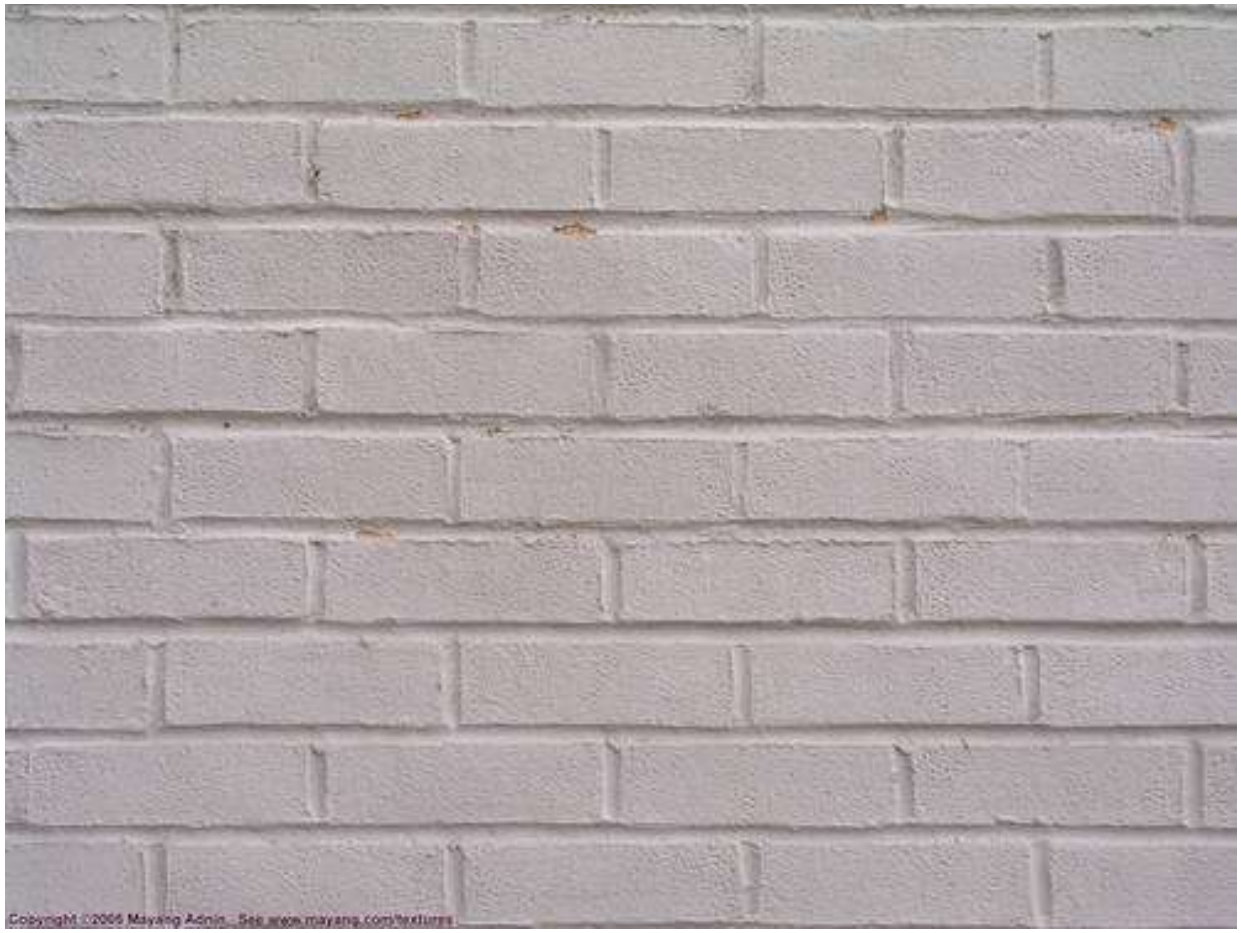
  - the image coordinates depend highly non-linearly on the world coordinates

Perspective projection

- this is the reason why we see this

Perspective projection

- instead of this



Perspective projection

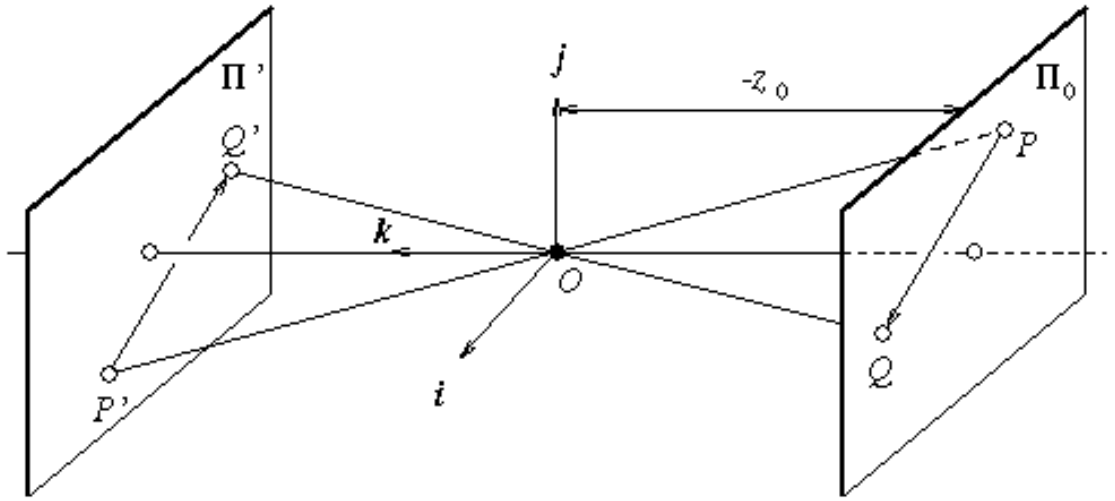
- since the size of the wall is constant
- far away (large z) distances appear to shrunk in the image
- in many cases, this non-linearity is too much to handle
- we look for approximations

Affine projection

- consider a plane parallel to the image plane
  - this plane has equation $z = C$ and the projection equation becomes

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \frac{f}{C}\begin{pmatrix} x \\ y \end{pmatrix} = m\begin{pmatrix} x \\ y \end{pmatrix}, \quad m = \frac{f}{C}$$

  - image coordinates are simply a re-scaling of the 3D coordinates



Affine projection

- scaling:
  - if m <1 image points are closer than 3D points,
  - else they are further away
  - this can be seen by noting that, for $P=(x_p,y_p)$, $Q=(x_q,y_q)$

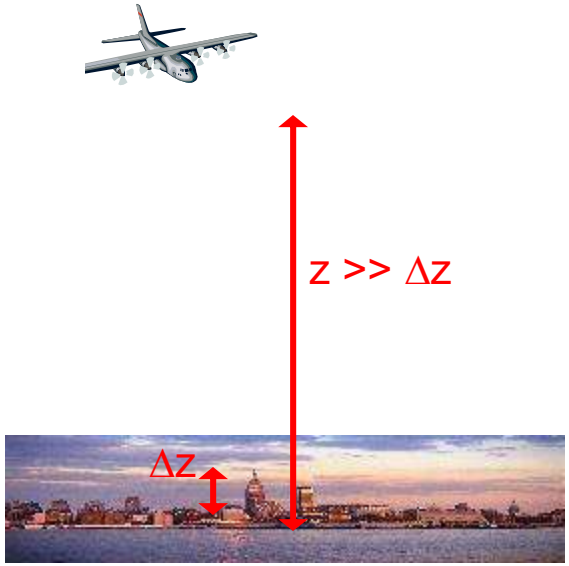
$$d(P',Q') = \sqrt{\left(x'_p - x'_q\right)^2 + \left(y'_p - y'_q\right)^2}$$
$$= \sqrt{m^2\left(x_p - x_q\right)^2 + m^2\left(y_p - y_q\right)^2}$$
$$= |m|\ d(P,Q)$$

  - this is also captured by the relation through a scaling matrix

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Affine projection

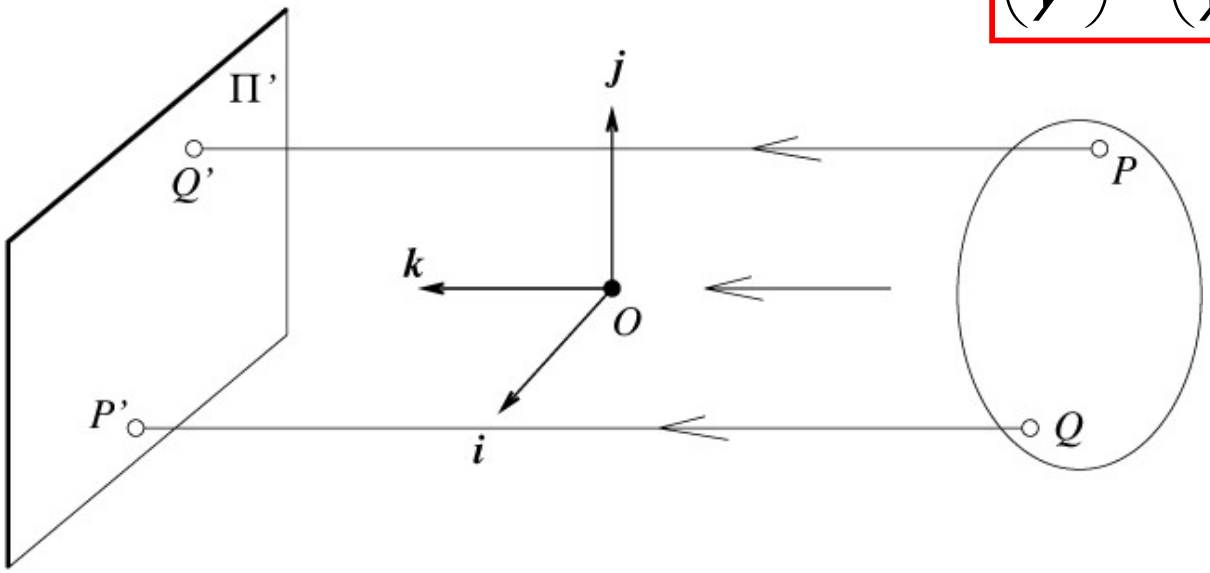
- when can we use this approximation?
  - we are assuming z constant
  - this is acceptable if the variation of depth in the scene is much smaller than the average depth
  - e.g. an airplane taking aerial photos



Orthographic projection

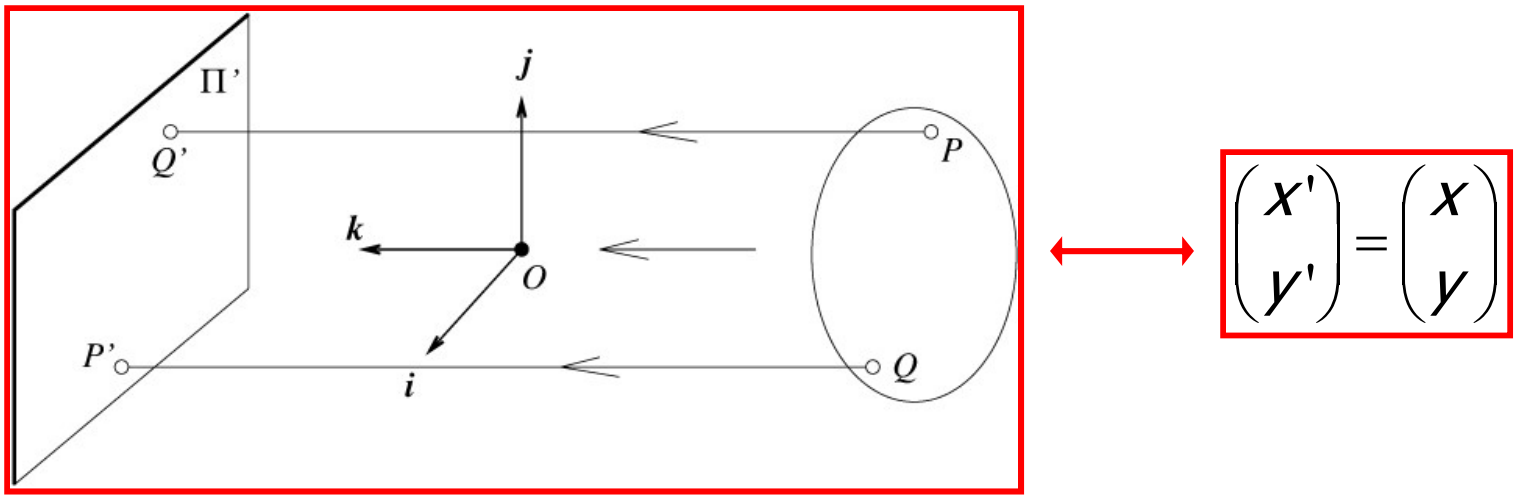
- if the camera is always at (approximately) the same distance from the scene:
  - m only contributes a change of scale that we do not care much about (e.g. measure in centimeters vs meters)
  - it is common to normalize to $m = 1$
  - this is orthographic projection

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}$$



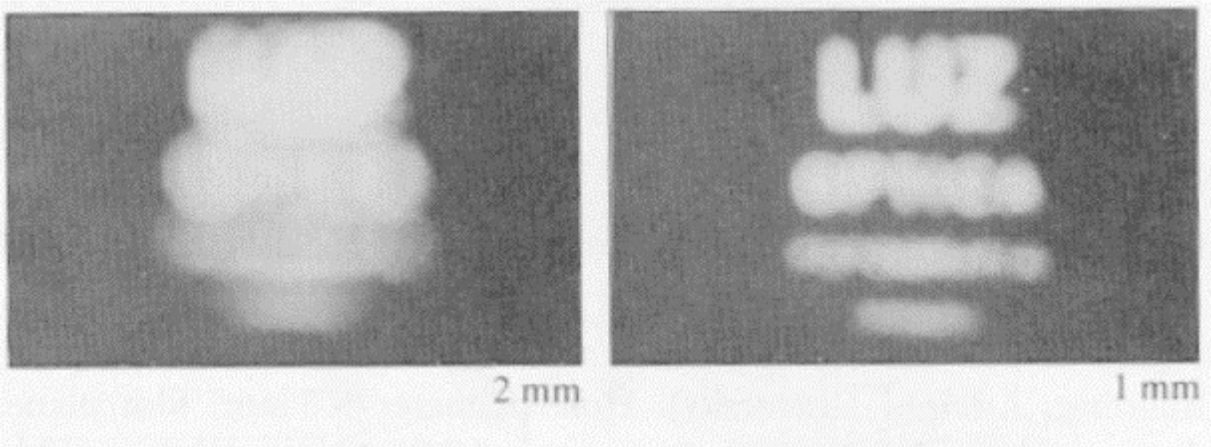
Orthographic projection

- this is, of course, very convenient:
  - "image coordinates = world coordinates"
  - there are not that many scenarios in which it is a good approximation
  - nevertheless can be a good model for a preliminary solution
  - which is then refined with a more complicated model



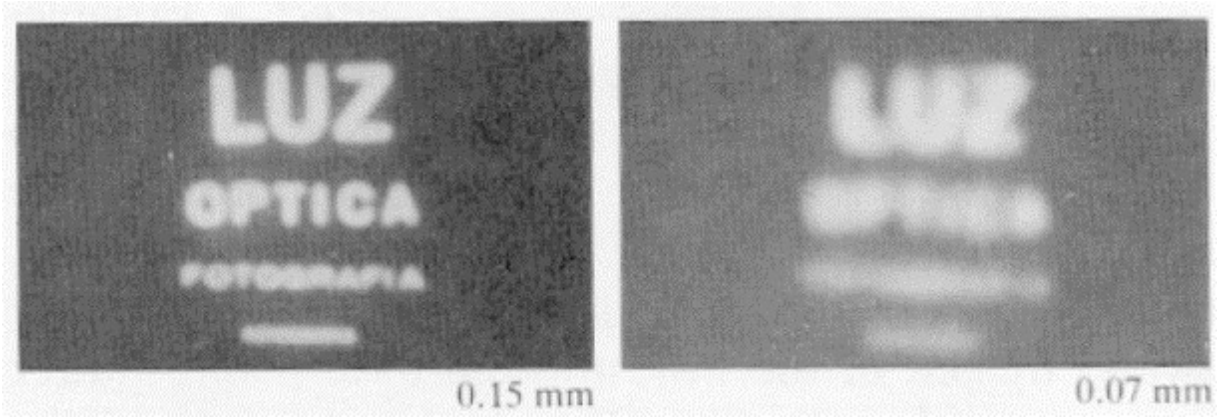
Lenses

- so far we have assumed the pinhole camera
- in practice we cannot really build such a camera and obtain decent quality
- problems:
  - when pinhole is too big
  - many directions are averaged, blurring the image



Lenses

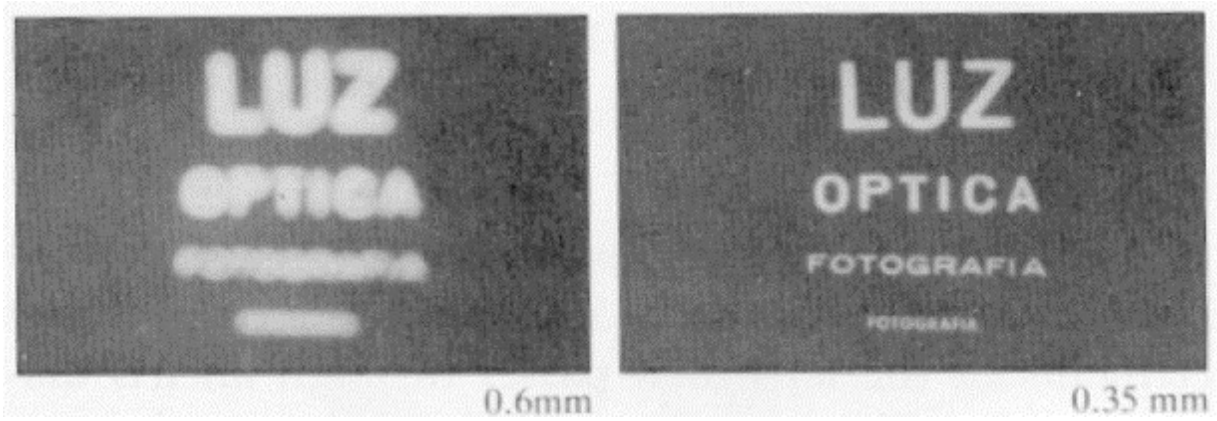
- pinhole problems:
  - if the pinhole is too small
  - we have diffraction effects which also blur the image



- there is a correct pinhole size from an image distortion point of view, but that introduces other problems

Lenses

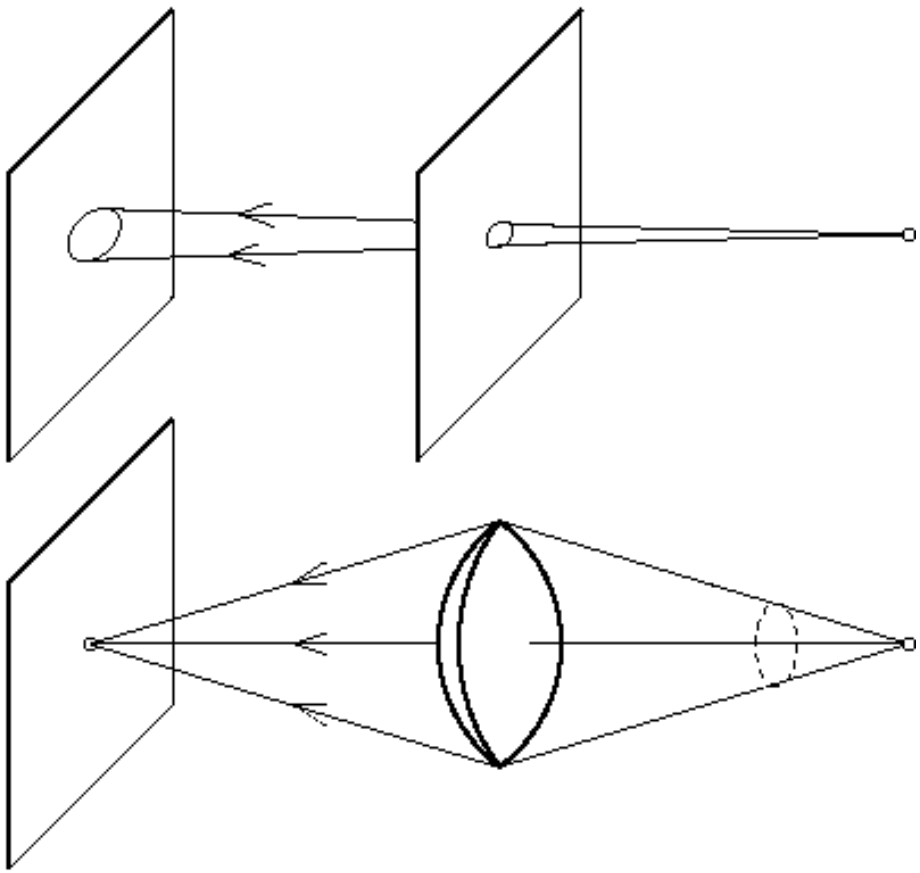
- pinhole problems:
  - for the "correct" pinhole size
  - we cannot get enough light in the camera to sufficiently excite the recording material
  - generally, pinhole cameras are *dark*,
  - a very small set of rays from a particular point hits the screen



  - this is the reason why we need camera lenses

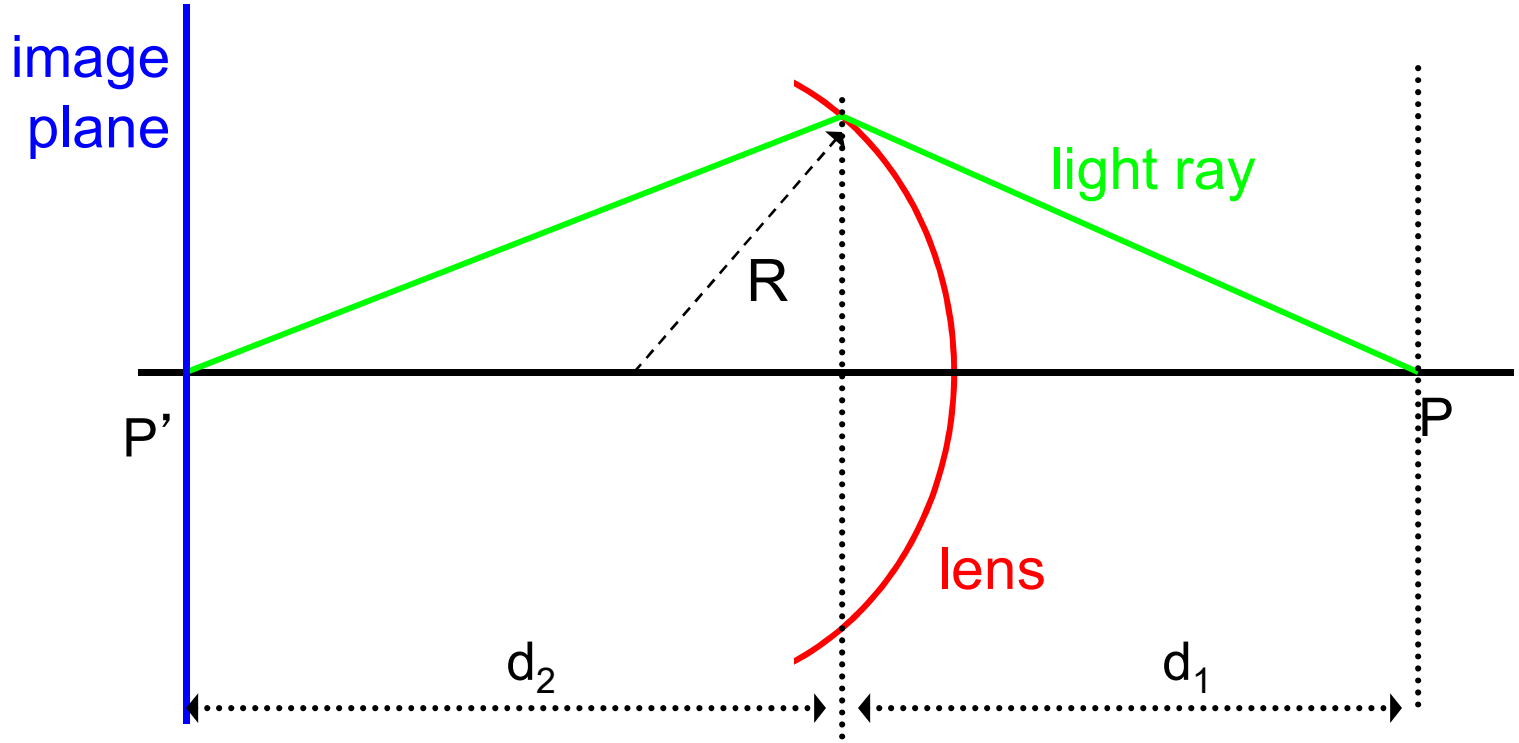
Lenses

- the basic idea is:
  - lets make the aperture bigger so that we can have many rays of light into the camera
  - to avoid blurring we need to concentrate all the rays that start in the same 3D point
  - so that they end up on the same image plane point



Lenses

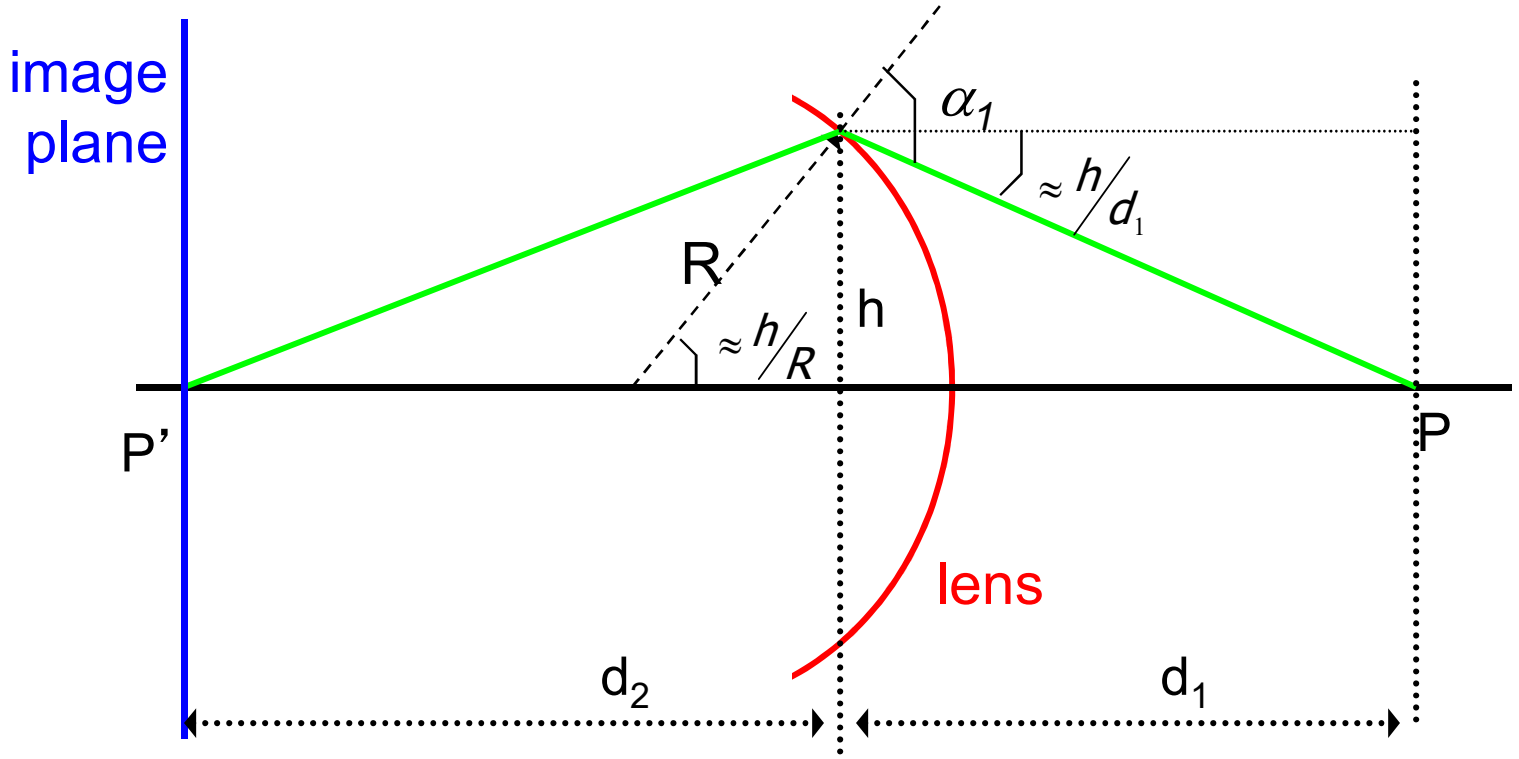
- the geometry is as follows



R: radius of curvature of the lens
d_1: distance from 3D point to lens
d_2: distance to image plane

Lenses

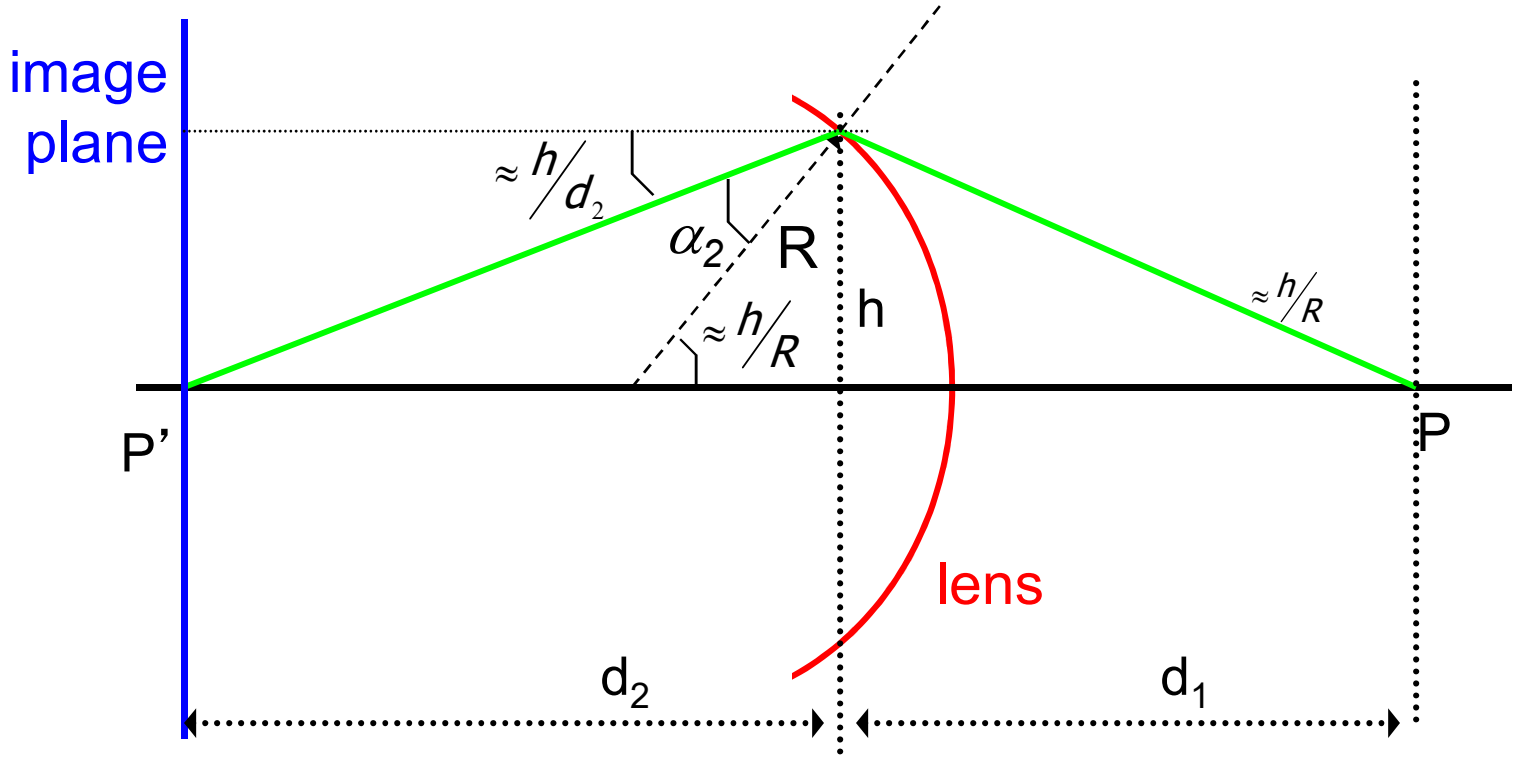
- we assume all angles are small (d_2, h are in microns):
  - remember trigonometry if α small: $\alpha = sin\ \alpha = tg\ \alpha$



$$\alpha_1 \approx \frac{h}{R} + \frac{h}{d_1}$$

Lenses

- we assume all angles are small (d_2, h are in microns):
  - $\alpha = \sin\alpha = tg\,\alpha$

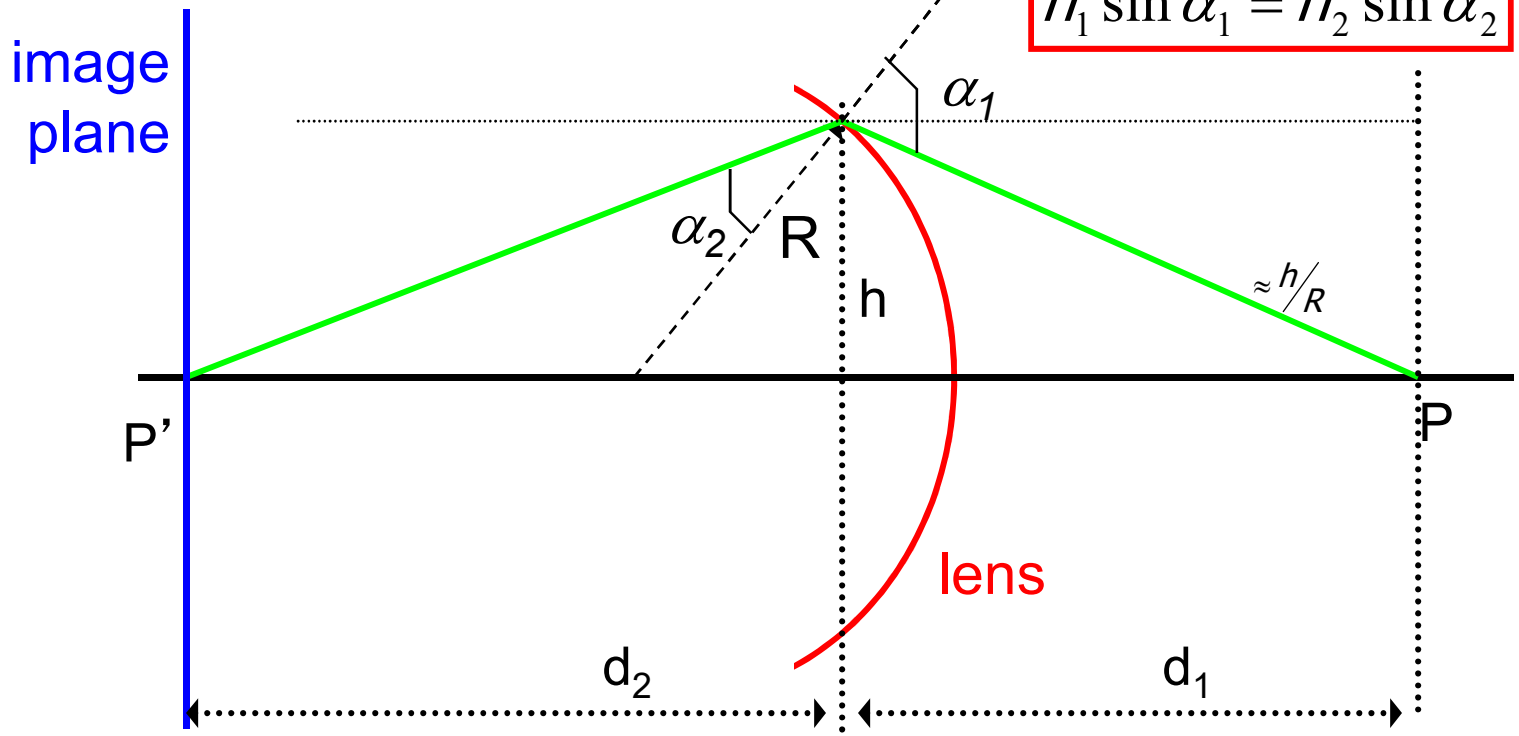


$$\alpha_2 \approx \frac{h}{R} - \frac{h}{d_2}$$

Lenses

- Snell's law
  - for light propagating between two media of indexes of refraction n_1 and n_2

$$n_1 \sin\alpha_1 = n_2 \sin\alpha_2$$

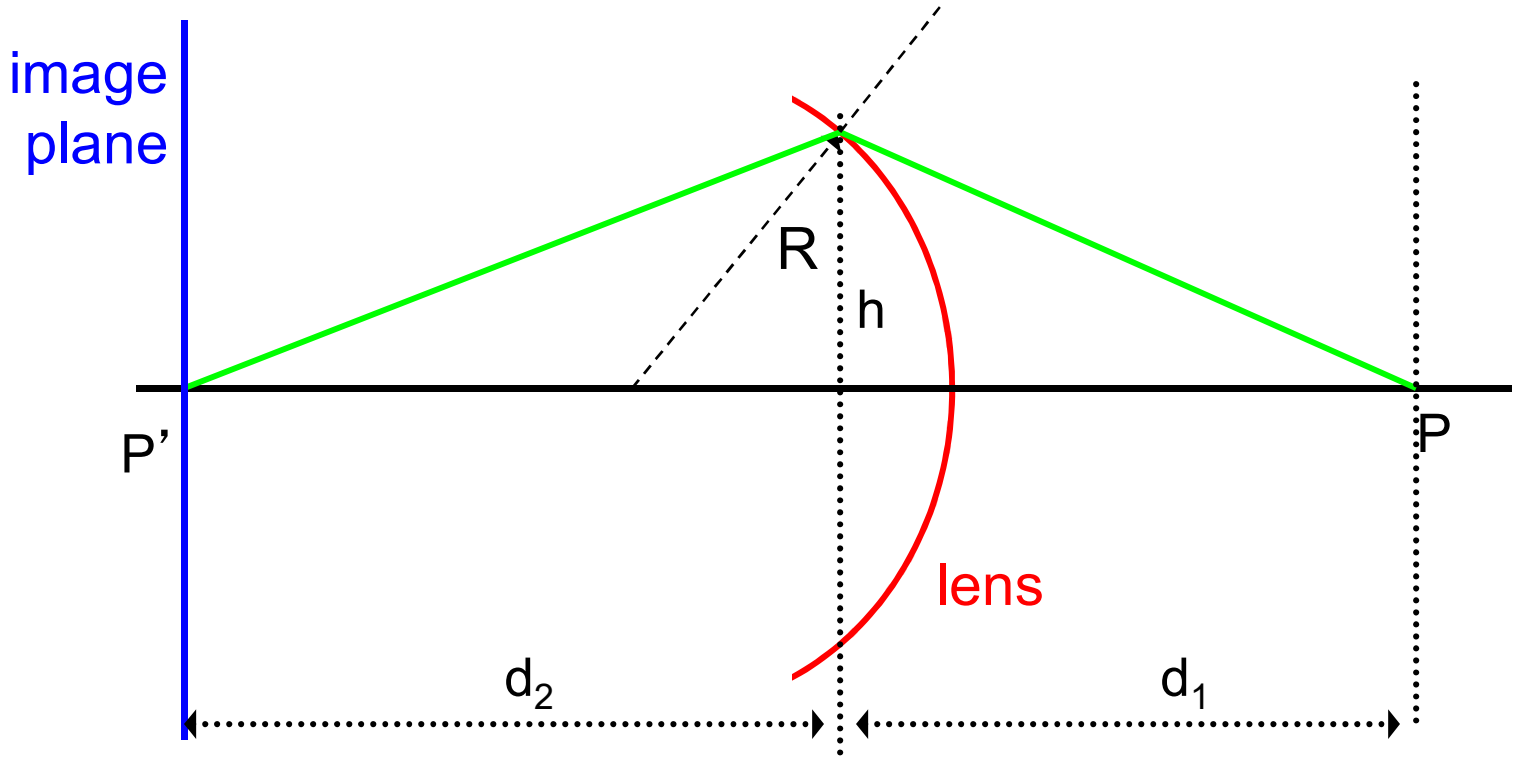


$$\alpha_1 \approx \frac{h}{R} + \frac{h}{d_1} \qquad \alpha_2 \approx \frac{h}{R} - \frac{h}{d_2}$$

$$\Rightarrow n_1 h\left(\frac{1}{R} + \frac{1}{d_1}\right) \approx n_2 h\left(\frac{1}{R} - \frac{1}{d_2}\right)$$

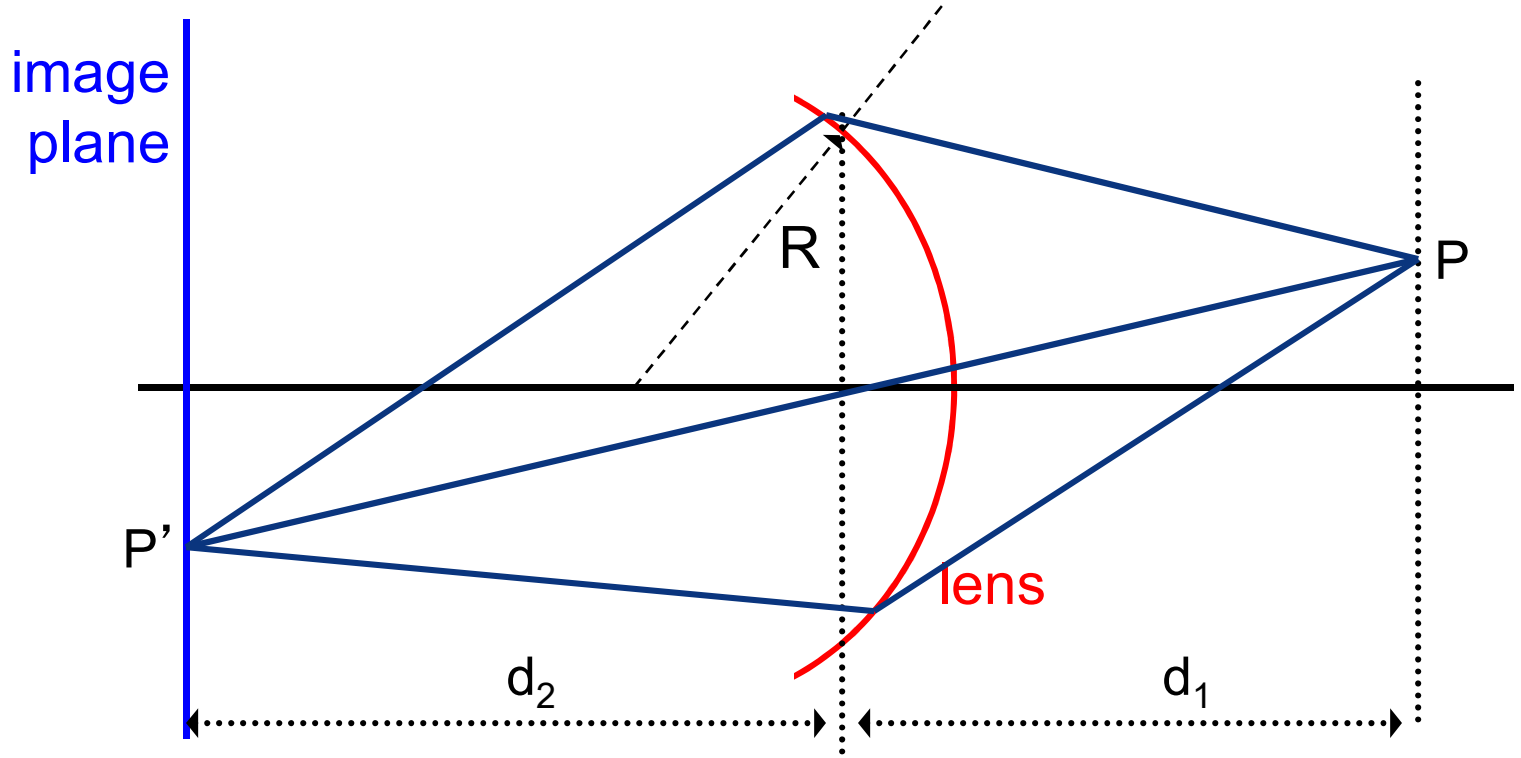
Lenses

- which means that $$\frac{1}{d_1} \approx \frac{1}{n_1}\left(\frac{n_2 - n_1}{R} - \frac{n_2}{d_2}\right)$$
  - given distance d_2, we can compute distance d_1 of the 3D point



Lenses

- which means that $$\frac{1}{d_1} \approx \frac{1}{n_1}\left(\frac{n_2 - n_1}{R} - \frac{n_2}{d_2}\right)$$
  - note that it does not depend on the vertical position of P
  - we can show that it holds for all rays that start in the plane of P



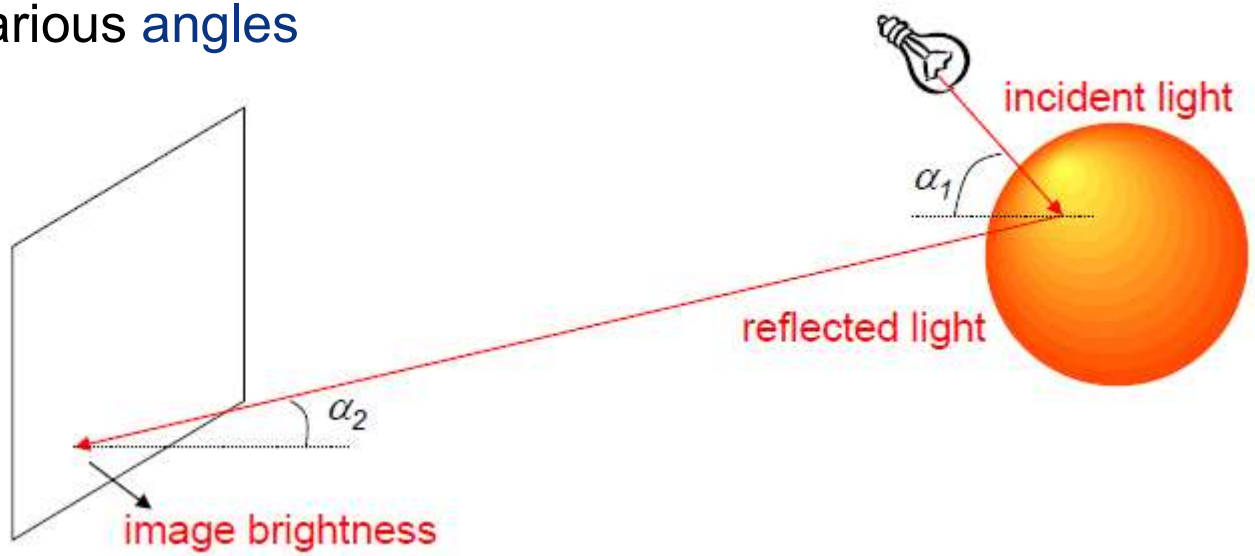
Lenses

- note that, in general,
  - we can only have in focus objects that are in a certain depth range
  - this is why the background is sometimes out of focus on photographs

  - by controlling the focus you are effectively changing the plane of the rays that converge on the image plane without blur

Light

- What is pixel brightness or image intensity?
- Three factors:
  - Lighting of the scene
  - The reflectance properties of the material
  - Various angles



Radiance

- To study light the first important concept is radiance
  - Appropriate unit for measuring distribution of light

- Definition: Radiance is
  - Power (energy/unit time) traveling at x in direction V
  - Per unit area perpendicular to V
  - Per unit solid angle

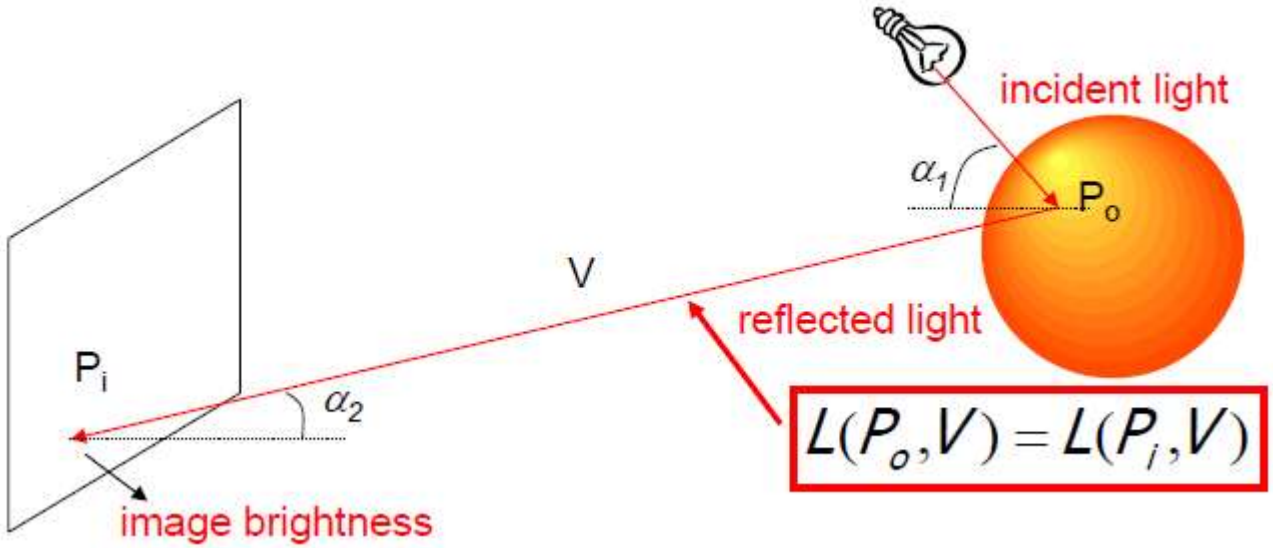
- It is measured in
  - Watts/square metter x steradian (w x m^{-2} x sr^{-1})
  - (steradian = radian squared)

$$L(x,V)$$

- It follows that it is a function of a point x and a direction V

Radiance (contd.)

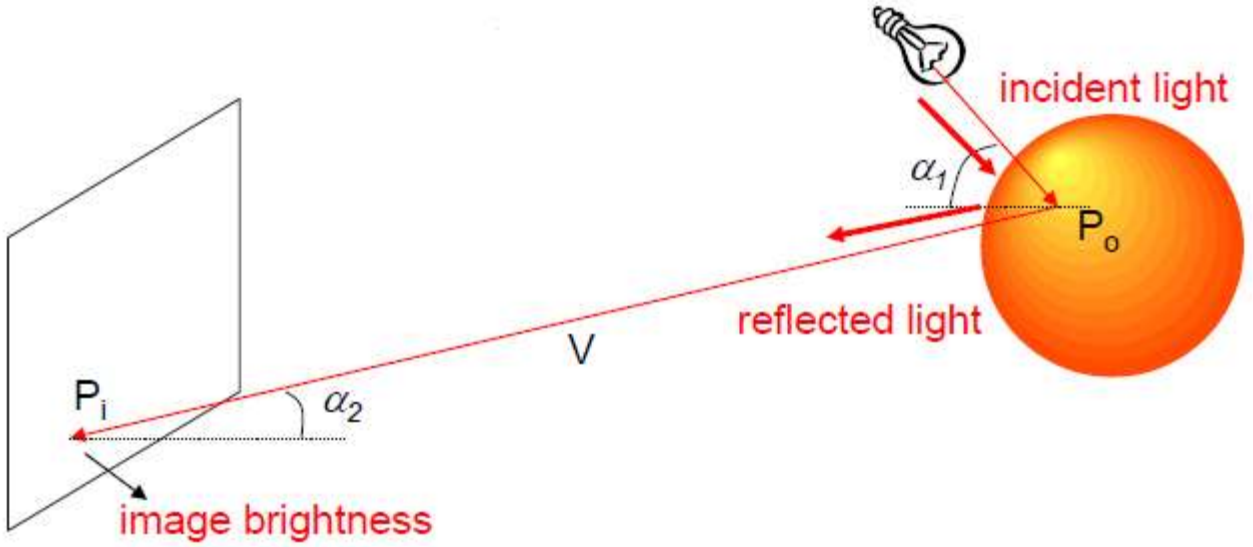
- Important property, in a lossless medium:
  - Whatever radiance is emitted by the object at P_o
  - Is the radiance received by the image at P_i



- I.e. In a lossless medium, radiance is constant along straight lines

Light

- So what is the relation between:
  - The illumination that reaches the object
  - And the reflected light?

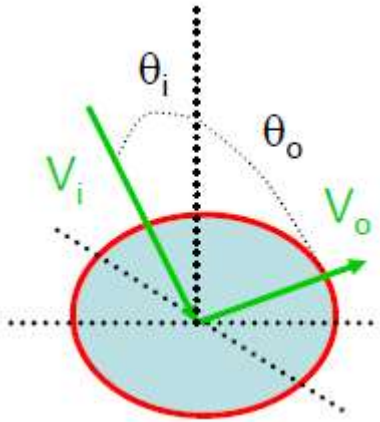


- This is measured by the Bidirectional Reflectance Distribution Function (BRDF)

BRDF

- Is the ratio of energy in outgoing direction (V_o) to incoming direction (V_i)

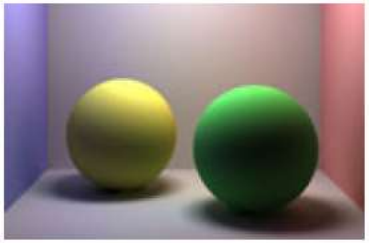
$$\rho_{bd}(P, V_i, V_o)$$



- Important property (Helmoltz reciprocity)
  - BRDF is symmetric

$$\rho_{bd}(P, V_i, V_o) = \rho_{bd}(P, V_o, V_i)$$

- But we can do even simpler than this
  - For Lambertian surfaces, the BRDF does not depend on the direction at all
  - These surfaces reflect light equally in all directions



$$\rho_{bd}(P, V_i, V_o) = \rho_{bd}(P)$$

Albedo

- For this particular case the surface is described by its albedo

$$\rho(P)$$

- Note that:
  - Most surfaces are not lambertian
  - But Lambertian assumption makes equations a lot easier
  - Commonly used in practice, even though Lambertian objects do not appear realistic (not good for graphics)

Albedo

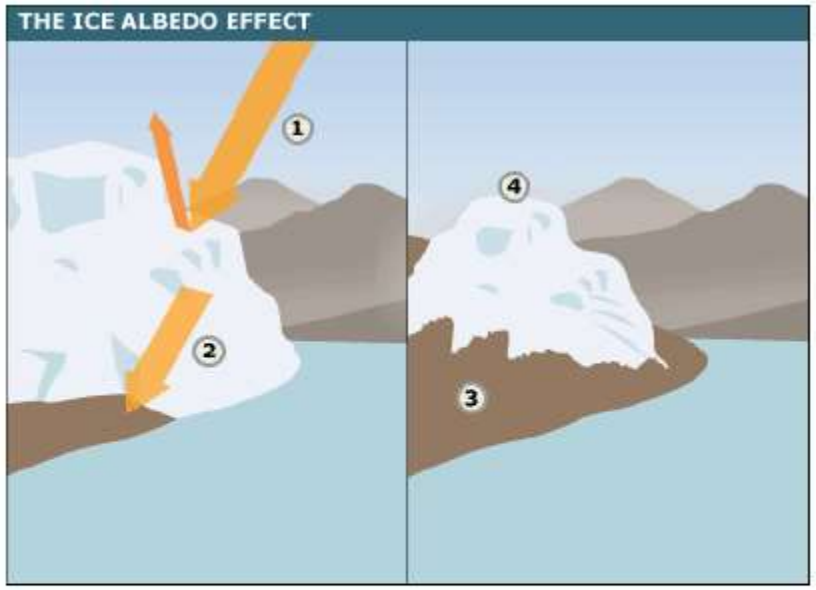
- Albedo is percentage of light reflected by an object
- it depends on the color and material properties of the object
- light colors reflect more light (that is why you should wear white in the desert)
- this turns out to have major consequences for object temperature
- e.g., it is one of the main justifications for global warming

Albedo and global warming

- snow turns out to have the largest possible albedo, and reflects almost 100% of the light
- most other objects absorb light, and heat up

Albedo and global warming

- by reflecting most of the incident light, the polar cap cools off the planet
- as ice melts, less light is reflected
- the planet warms up, more ice melts, etc.
- this is one of the main reasons for global warming



1 Light colored ice reflects back the Sun's energy efficiently.
2 Exposed land is darker colored and absorbs more energy.
3 As the ice melts, more land is exposed. This absorbs more heat, melting more ice.
4 The altitude of the melting ice is reduced so it becomes harder for new ice to form.

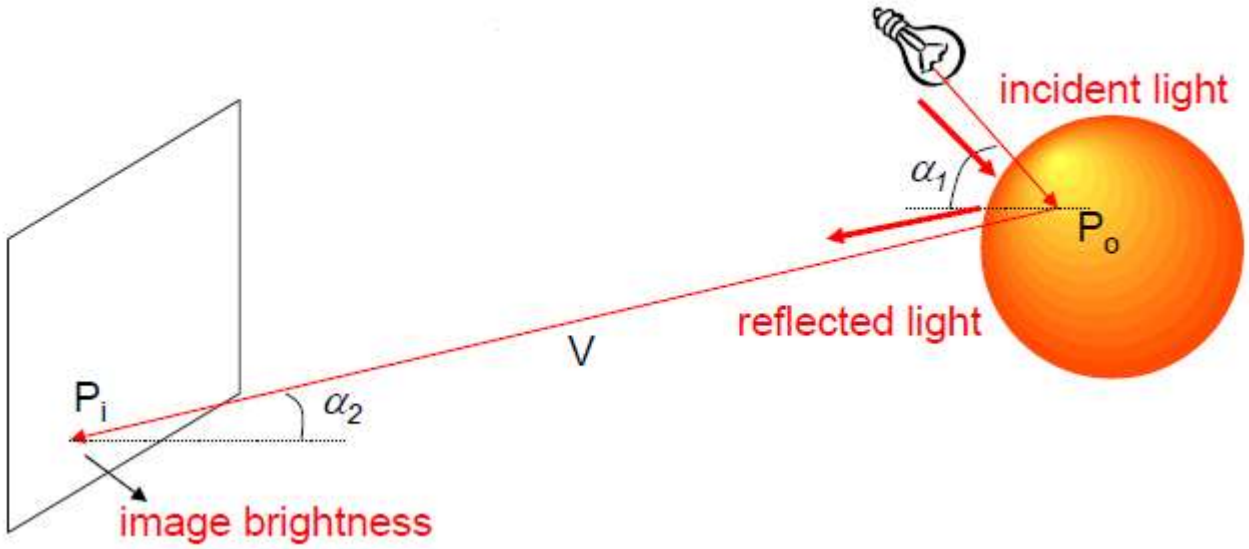
Albedo and parking lots

- it turns out that increasing reflections are useful in many other ways
- for example, it can save a lot of lighting (energy)

- an example of how paving parking lots with pavement of higher reflectance can make a difference

Angles

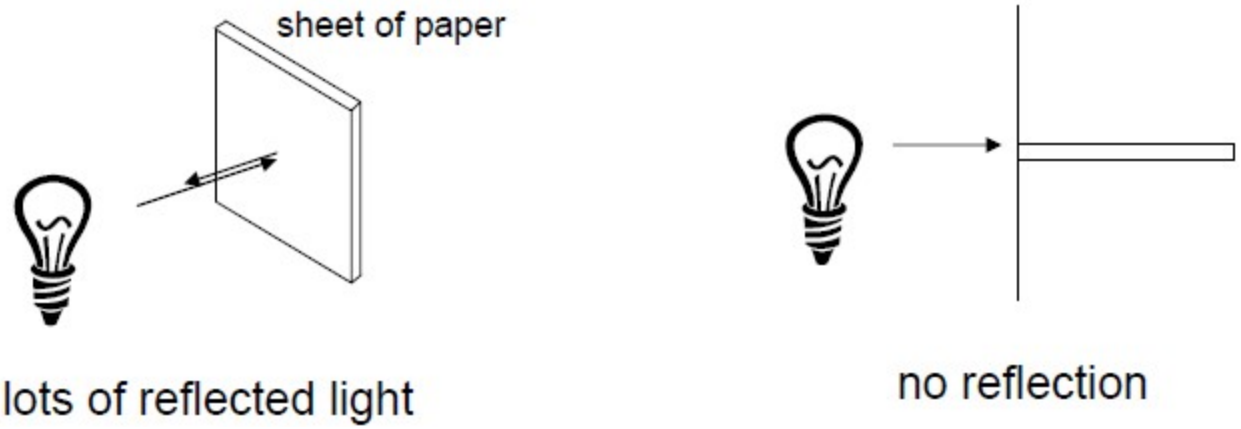
- is the object albedo the only factor that matters for how much light it reflects?



- no, the angle at which the light is incident also matters

Light

- this is easy to see, consider the following experiment



- energy absorbed by object depends on its surface area
- this varies with the incident angle
- concept that captures this dependence on angles is that of foreshortening

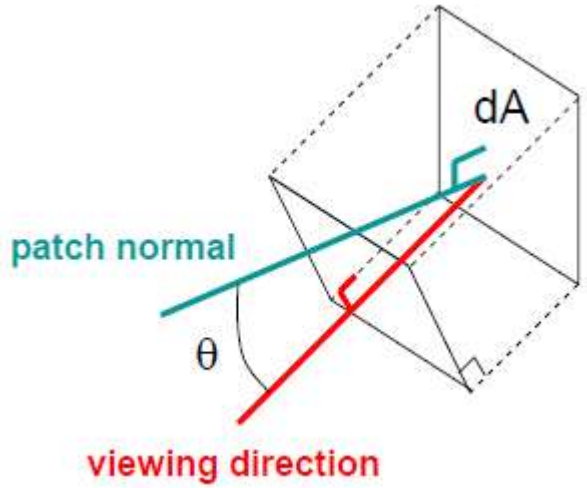
Light

- foreshortening: very important concept
  - tilted surface looks smaller than when seen at 90^{o}
  - best understood by example

  - if I show you a tilted person it looks smaller than when you view you at 90^{o}

Light

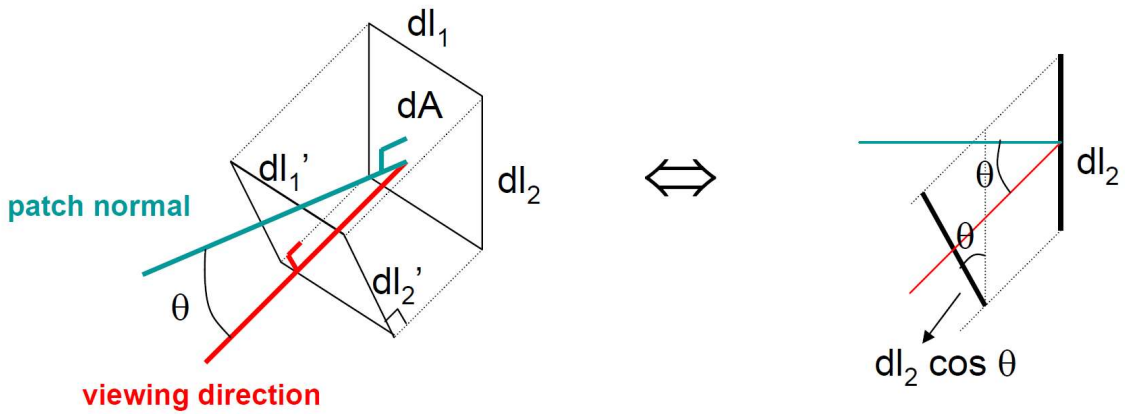
- what is the foreshortened area for a patch of area dA?



- it depends on the angle between:
  - the normal to the patch
  - viewing direction
- for a known area dA we can actually compute the foreshortening factor

Light

- this is easy for a simple case



- foreshortened area is
  - $dA' = dl_1' \, dl_2' = dl_1 \, dl_2 \cos\theta = dA \cos\theta$
- it can be shown that this holds for a patch of any shape:
  - foreshortened area = area x cos (angle between viewing direction and surface normal)

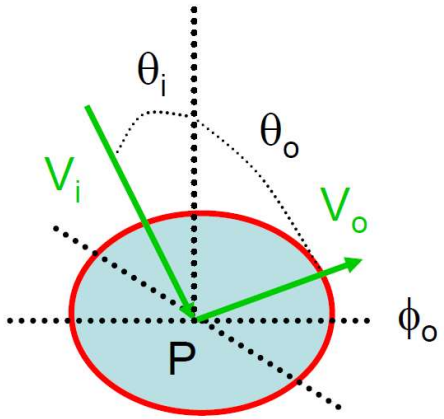
Lambertian surfaces

- putting everything together,
  - we have an equation for the light reflected by an object
  - Assuming surface reflects equally in all directions (Lambertian)
  - outgoing radiance is

$$L(P,V_o) = \rho(P)L(P,V_i)\cos\theta_i, \forall V_o$$

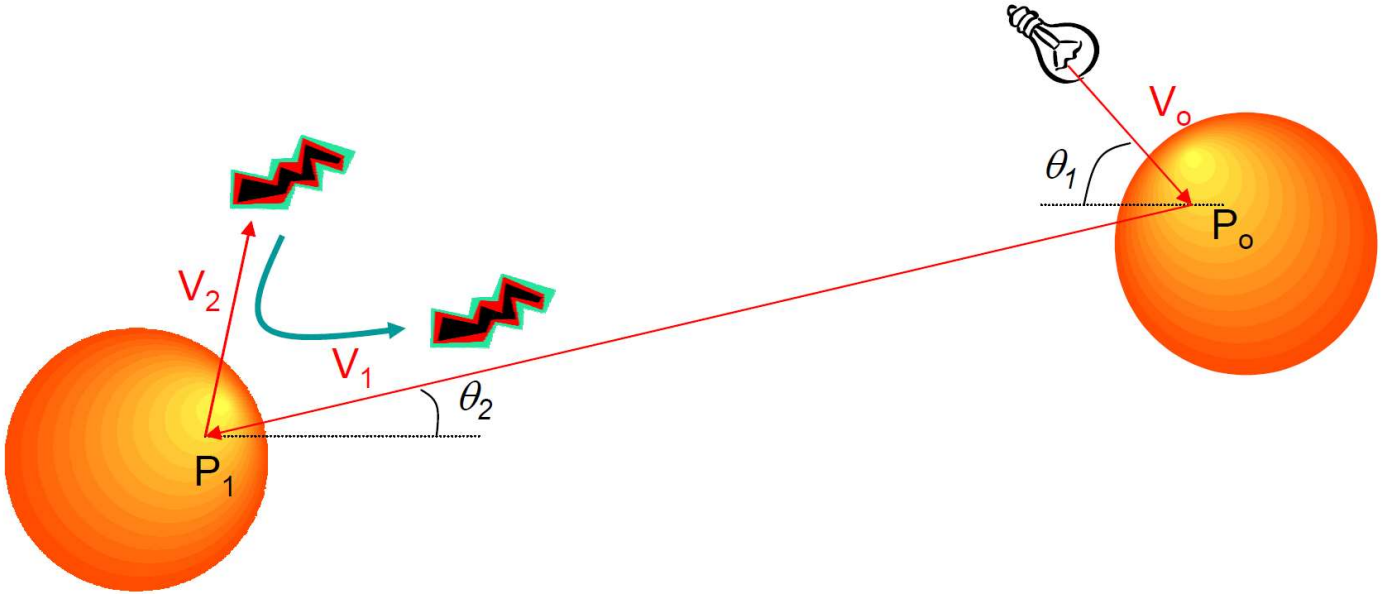
- 'light reflected at point P in direction V_o' = 'albedo at P' x 'incident light from direction V_i' x 'cos(normal, view)'

- note that:
  - it holds for any outgoing direction V_o
  - is a function of V_i
  - if there are multiple incoming directions, we have to integrate over V_i



Lambertian surfaces

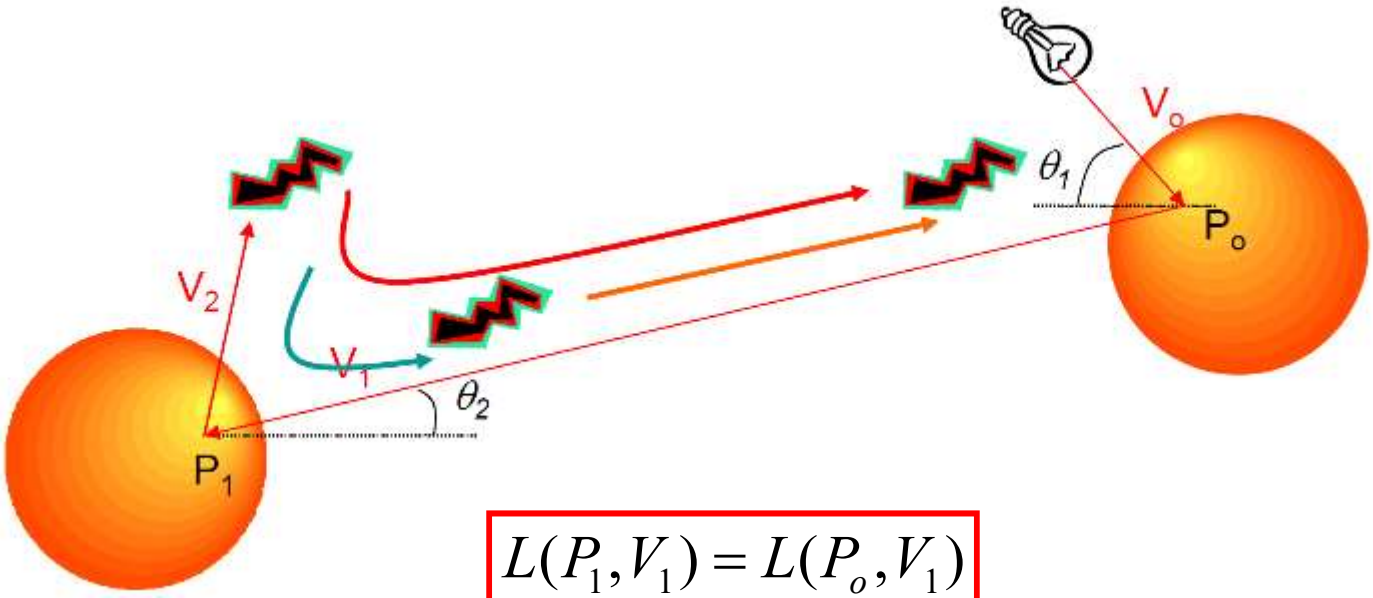
- This allows us to propagate light throughout a scene



$$L(P_1, V_2) = L(P_1, V_1)\rho(P_1)\cos\theta_2, \forall V_2$$

Lambertian surfaces

- Using constancy of radiance along straight lines



$$L(P_1, V_1) = L(P_o, V_1)$$

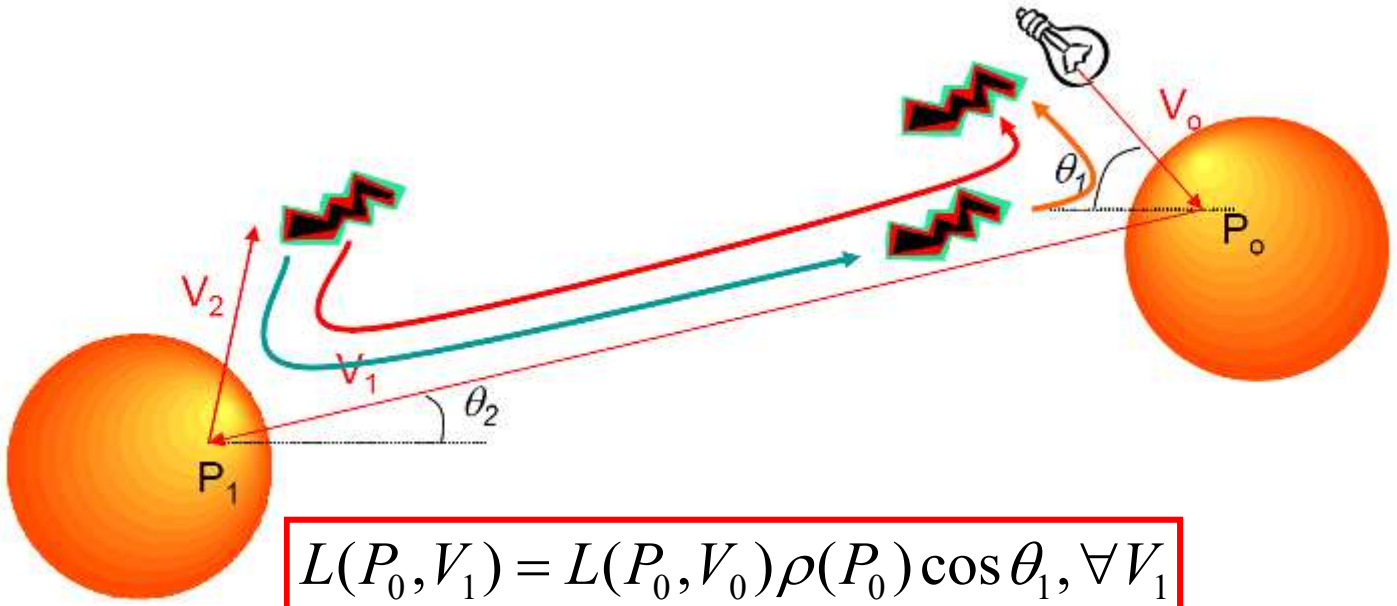
$$L(P_1, V_2) = L(P_1, V_1)\rho(P_1)\cos\theta_2, \forall V_2$$

$$\Updownarrow$$

$$L(P_1, V_2) = L(P_0, V_1)\rho(P_1)\cos\theta_2, \forall V_2$$

Lambertian surfaces

- Using reflection equation again



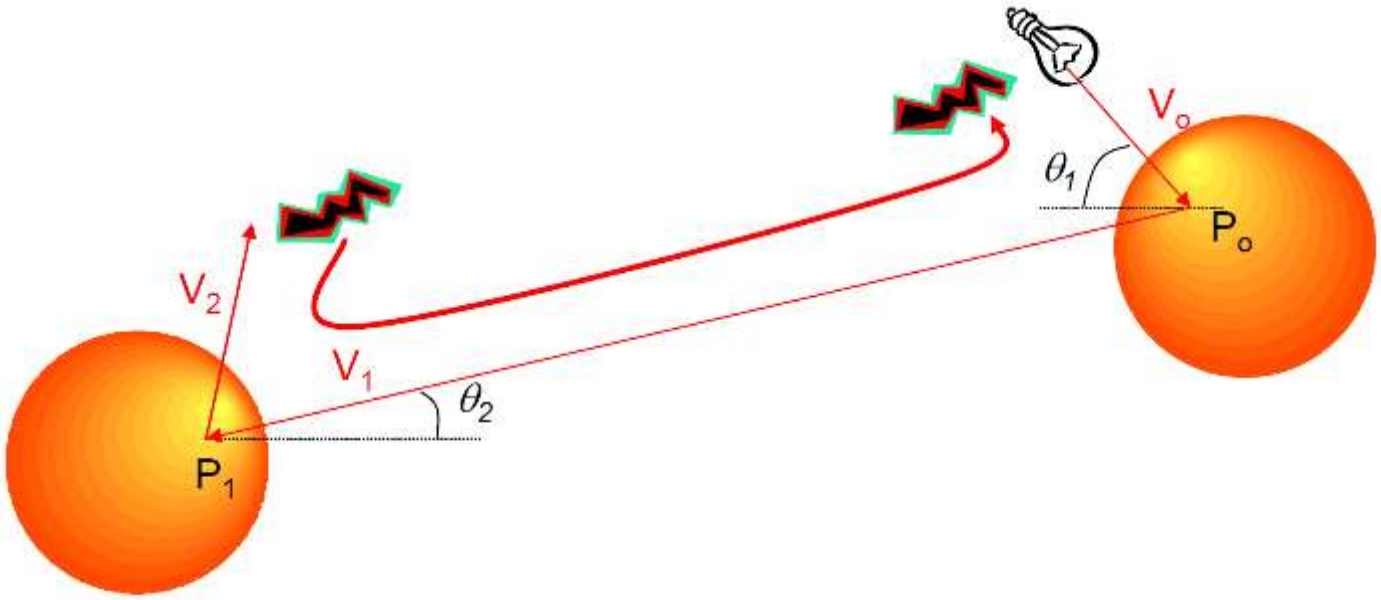
$$L(P_0, V_1) = L(P_0, V_0)\rho(P_0)\cos\theta_1, \forall V_1$$

$$L(P_1, V_2) = L(P_0, V_1)\rho(P_1)\cos\theta_2, \forall V_2$$

$$L(P_1, V_2) = L(P_0, V_0)\rho(P_0)\cos\theta_1\rho(P_1)\cos\theta_2$$

Lambertian surfaces

- Finally the rule for any number of bounces



$$L(P_n, V) = L(P_0, V_0)\left[\prod_{i=0}^{n} \rho(P_i)\right]\left[\prod_{i=1}^{n+1} \cos\theta_i\right], \forall V$$

Lambertian surfaces

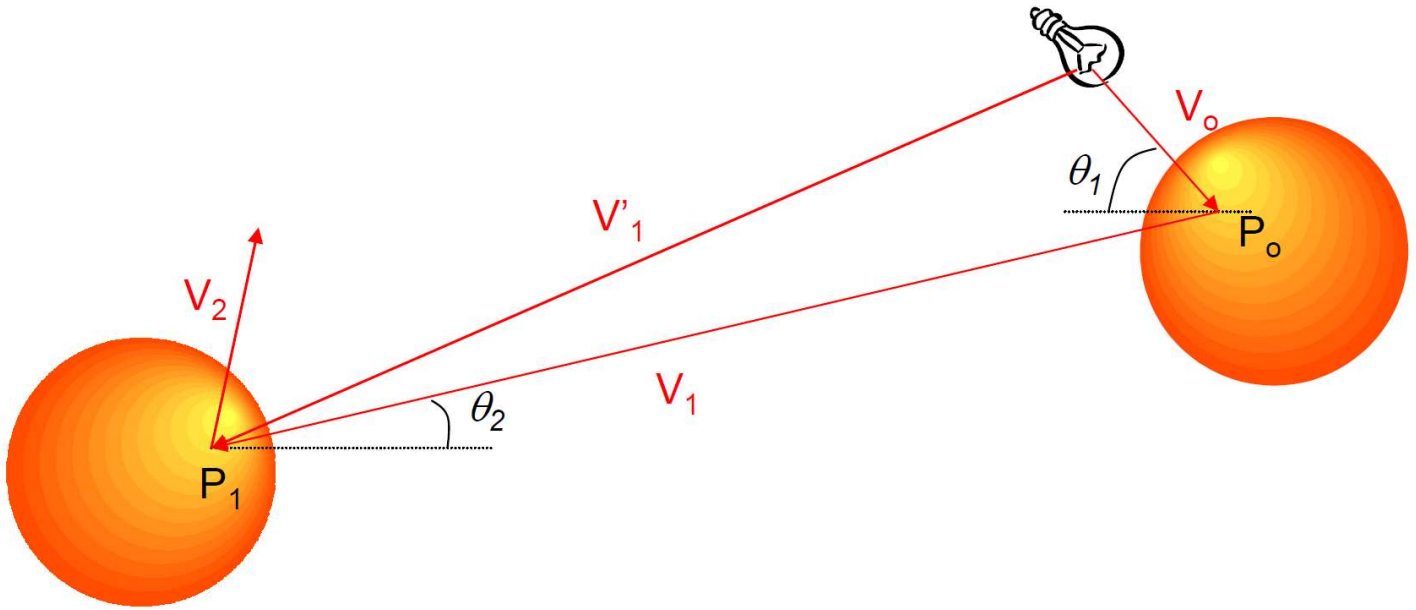
- Note that on

$$L(P_n, V) = L(P_0, V_0)\left[\prod_{i=0}^{n} \rho(P_i)\right]\left[\prod_{i=1}^{n+1} \cos\theta_i\right], \forall V$$

  - unless all cosines are close to 1, their product goes to zero quickly
  - this means that only light that arrives frontally (90 degrees) to all the bounces gets propagated very far
  - such an alignment is very unlikely
  - we don’t really have to worry about many bounces, hence the process becomes tractable

Lambertian surfaces

- But on the other hand
  - there are still various single-bounce paths
  - e.g. each source has a single bounce path to each non-shaded object
  - to deal with this we need to know more about light sources



Light sources

- Most common model is “point source at infinity”
  - assume all light comes from a single point
  - which is very far away from the scene

- Reasonable assumption for vision where one of two cases tend to hold
  - source is much smaller than the scene (e.g. a light-bulb)
  - source is very far away (e.g. the sun)

- hence, in general, relative to its size and the size of the scene the source can be considered distant

Color

- so far we have talked about geometry
  - where is a 3D point mapped into, in terms of image coordinates?
  - perspective projection

$$(x, y) = \frac{f}{Z}(X, Y)$$

- and light
  - what is the brightness of that image pixel
  - radiance of Lambertian surfaces, point source at infinity (E)

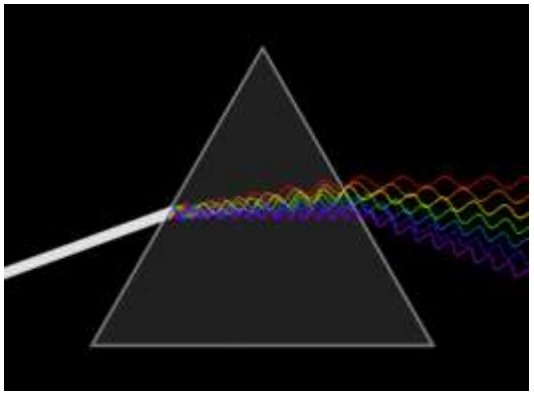
$$L = \frac{\rho_d}{\pi} E \cos\langle n, s \rangle$$

- Now we'll talk about the missing link: color

The appearance of colors

- To talk about color we need to know how it is perceived by people

- Principle of trichromacy:
  - it is possible to match almost all colors, using only three primary sources
  - this is not surprising given that we have three types of receptors
  - these are filters for some wavelengths
  - anything that falls out of the combination of the three primaries can't be seen by us

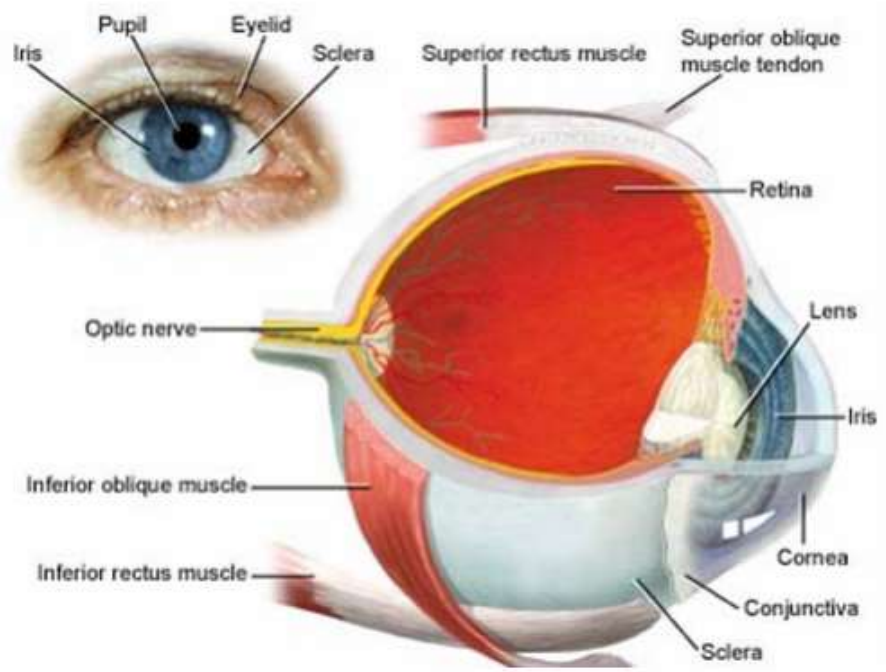


The principle of trichromacy

- Empirical evidence:
- Three primaries will work for most people if we allow subtractive matching
  - Exceptional people can match with two or only one primary.
  - This could be caused by a variety of deficiencies.
- Most people make the same matches.
  - There are some anomalous trichromats, who use three primaries but make different combinations to match.
- Color matching is linear: Grassman's laws (later)

Color

- how do we perceive color?
- color perception is the result of evolution

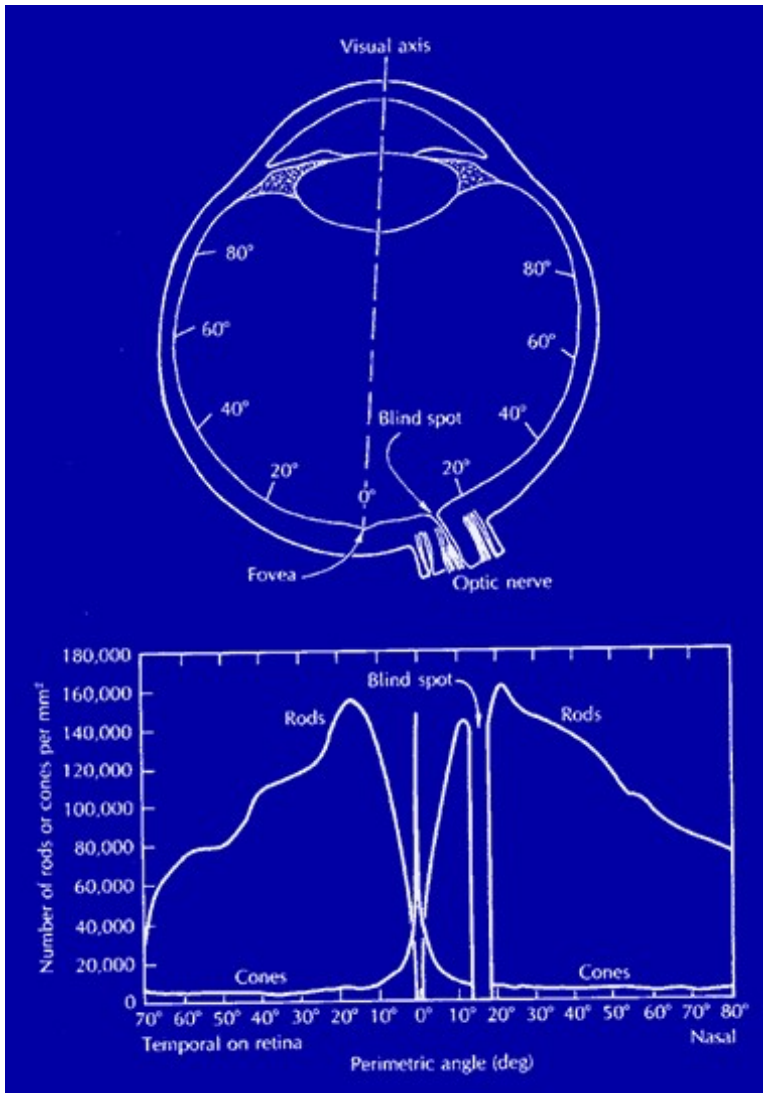


- let's look at the human eye:
  - light enters through pupil
    - this has the role of a lens
  - projects into the retina
    - this is the "camera plane"
  - retina covered by receptors (cones and rods) that transform light into electric pulses
  - sent to the brain through the optic nerve

Why three primaries?

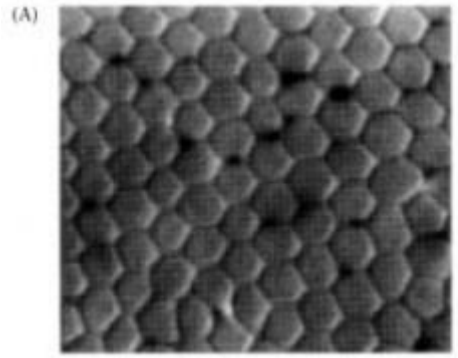
- Two types of receptors in the retina:
  - rods: responsible for vision at low light levels, do not mediate color vision
  - cones: are active at higher light levels, capable of color vision
- rods dominate in the periphery of the eye,
- the center is mostly composed of cones
- three types of cones:
  - S (respond to short wavelengths)
  - M (medium), and
  - L (long)

Rod vs cone density

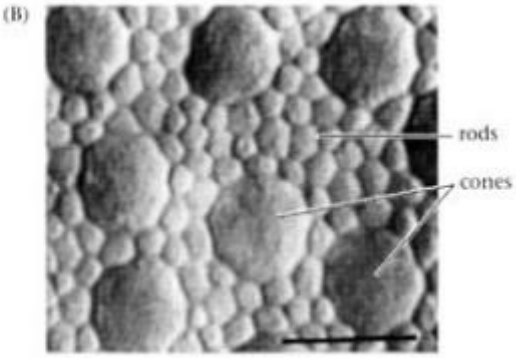


- very high concentration of cones in the center
- high resolution and color
- density decays very quickly

center

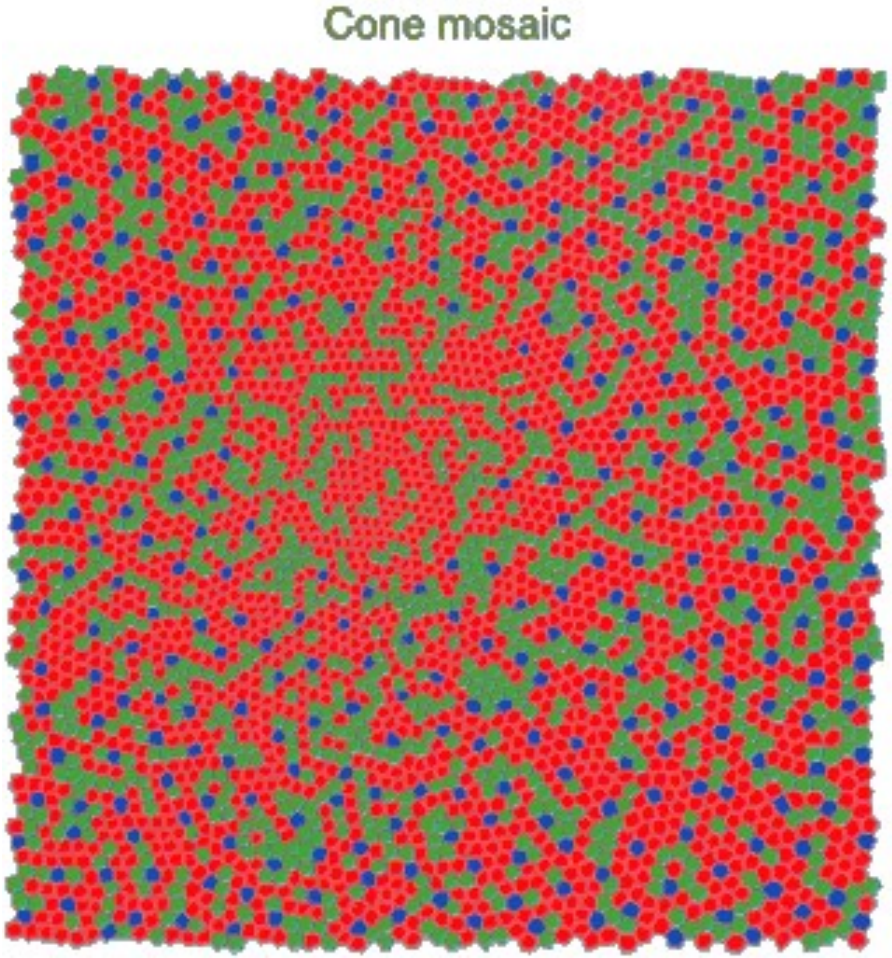


periphery

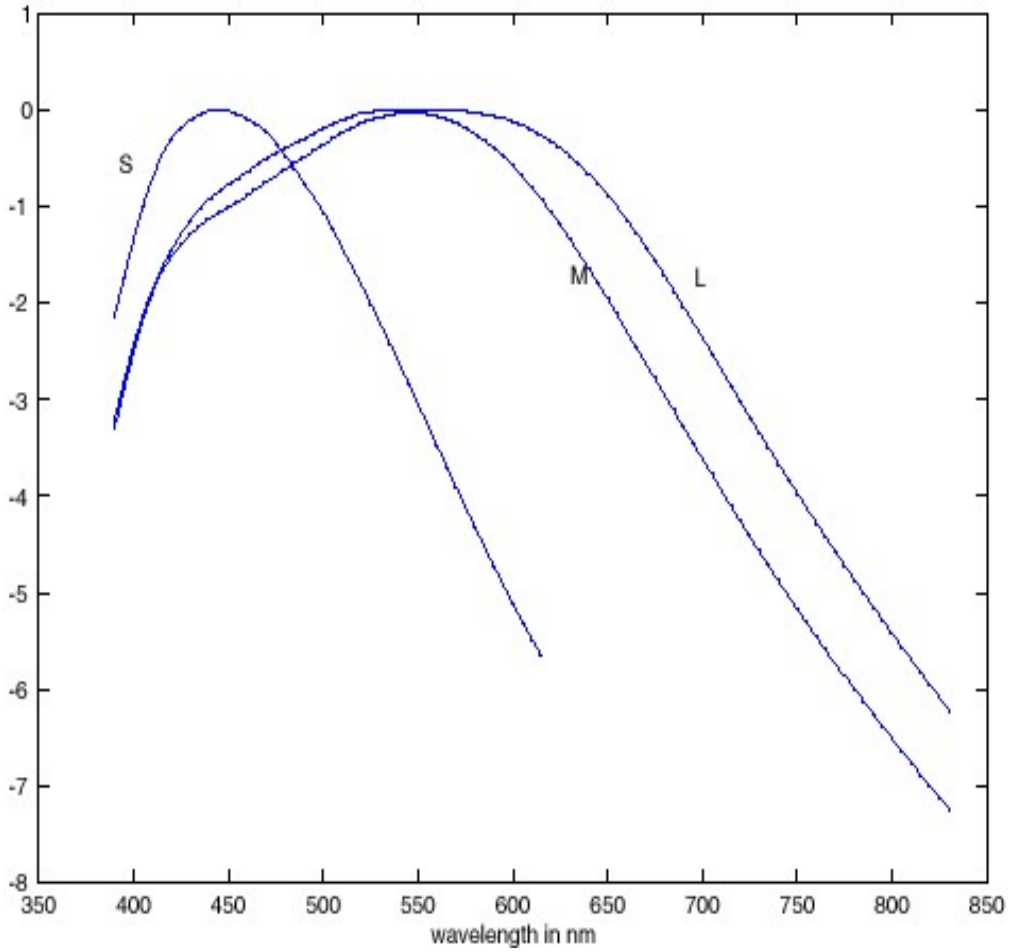


Color receptors and thricromacy

- trichromacy is justified: three types of cones, which vary in their sensitivity to light at different wavelengths
- picture shows a cone mosaic, spatial distribution of the different types of cones
- blue: S cones
  green: M cones
  red: L cones
- note the different densities



Color receptors



- relative sensitivity as a function of wavelength. S (for short) cone responds most strongly at short wavelengths; the M (for medium) at medium wavelengths and the L (for long) at long wavelengths.
- occasionally called B, G and R cones respectively, but that's misleading - you don't see red because your R cone is activated.
- explains 3 primaries

Radiometry for color

- All definitions become "per unit wavelength"
- All units become "per unit wavelength"
- All terms become "spectral"
- Radiance becomes spectral radiance $L^{\lambda}(x,\theta,\phi)$
  - watts per square meter per steradian per unit wavelength
  - radiance emitted in $[\lambda,\lambda+d\lambda]$ is

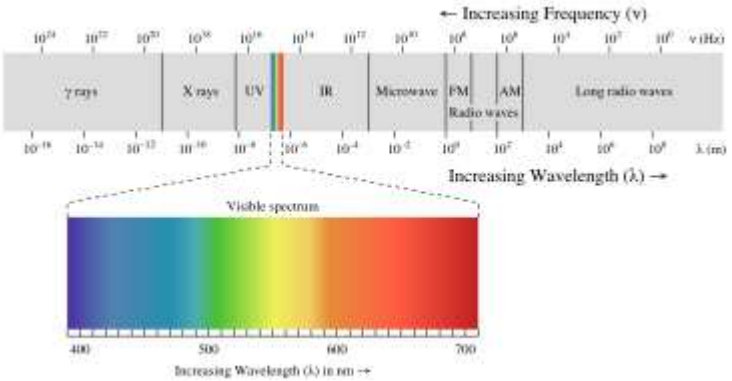
$$L^{\lambda}(x,\theta,\phi)d\lambda$$

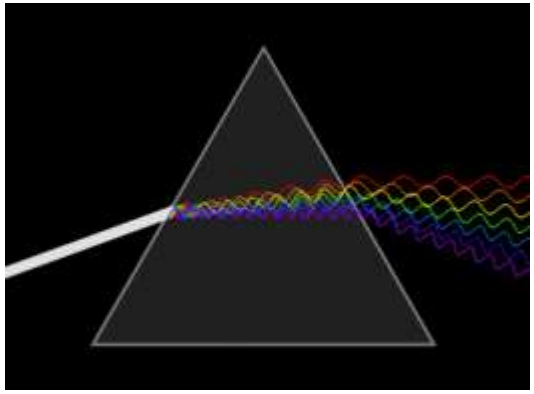
- Spectral radiosity, etc.
- Spectral BRDF

$$\rho_{bd}{}^{\lambda}(\theta_i,\phi_i,\theta_o,\phi_o) = \frac{L_o^{\lambda}(P,\theta_o,\phi_o)}{L_i^{\lambda}(P,\theta_i,\phi_i)\cos\theta_i d\omega}$$

Radiometry for color

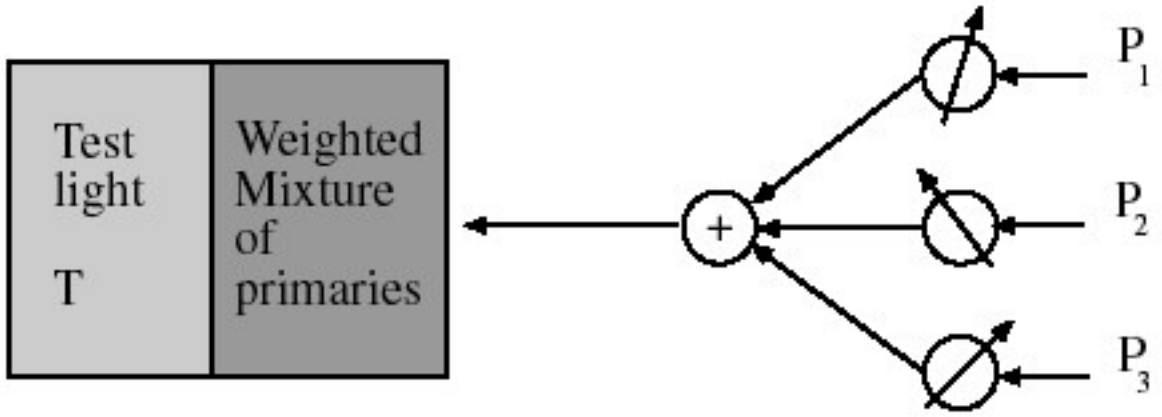
- the three types of cones are filters, specialized to certain wavelengths
- why does this happen?
- because:
  - the visible spectrum covers a range of wavelengths
  - radiance is a function of wavelength λ
    - e.g. if you put a color filter in front of your lens, only certain wavelengths go through
    - and all objects reflect light differently at different wavelengths
    - this is what makes them look like they have color





Color matching experiments

- Subject shown a split field:
  - one side shows the light whose color one wants to measure,
  - other a weighted mixture of primaries (fixed lights).



- Subject adjusts dials so as to make mixture equal to test

Color matching (contd.)

- Most colors can be represented as a mixture of P_1, P_2, P_3

$$M = a\,P_1 + b\,P_2 + c\,P_3$$

  where the = sign should be read as "matches"

- This is additive matching.
- Important because if two people who agree on P_1, P_2, P_3 need only supply (a, b, c) to describe a color.
- Some colors can't be matched like this: instead we need

$$M + a\,P_1 = b\,P_2 + c\,P_3$$

- This is subtractive matching.
- We can interpret it as (-a, b, c)

Grassman's Laws

1. mixture of coordinates matches the mixture of the lights

$$\left.\begin{aligned} T_1 &= a_1P_1 + b_1P_2 + c_1P_3 \\ T_2 &= a_2P_1 + b_2P_2 + c_2P_3 \end{aligned}\right\} T_1 + T_2 = (a_1 + a_2)P_1 + (b_1 + b_2)P_2 + (c_1 + c_2)P_3$$

2. equal coordinates means equal lights

$$\left.\begin{aligned} T_1 &= aP_1 + bP_2 + cP_3 \\ T_2 &= aP_1 + bP_2 + cP_3 \end{aligned}\right\} \; T_1 = T_2$$

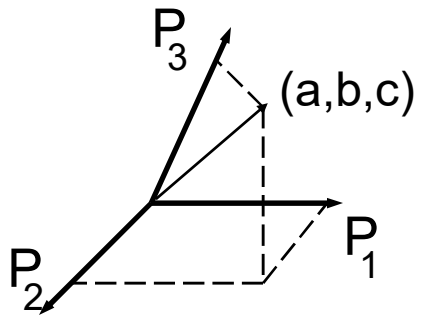
3. matching is linear

$$T_1 = aP_1 + bP_2 + cP_3 \Leftrightarrow kT_1 = kaP_1 + kbP_2 + kcP_{32}$$

- note: these statements are as true as any biological law.

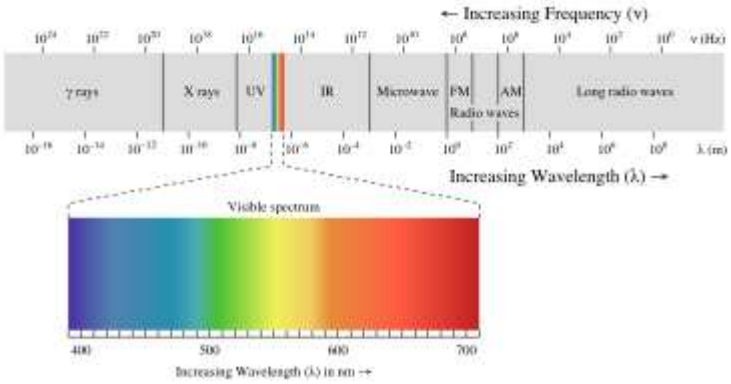
Linear color spaces

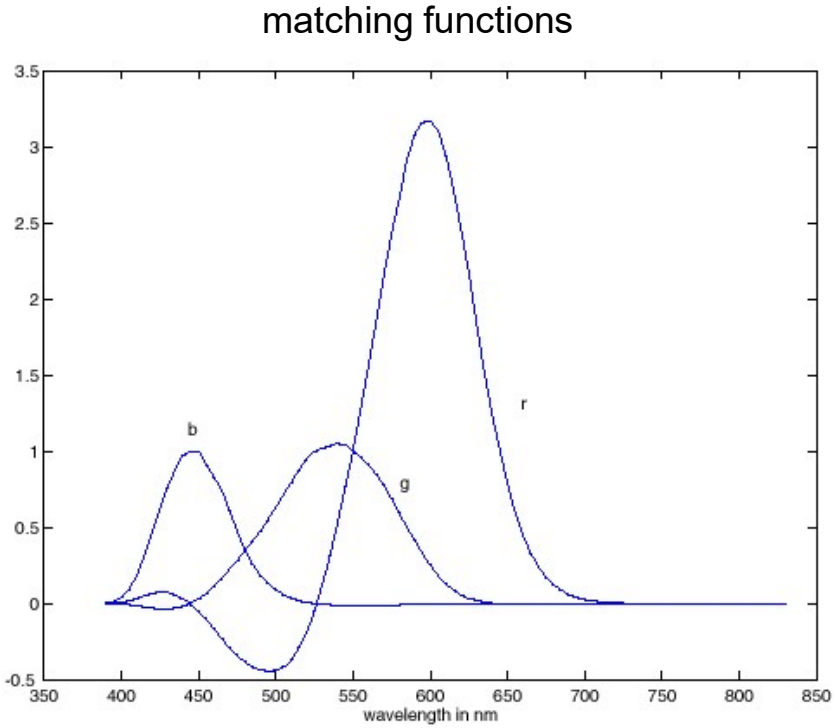
- Because color matching is linear in the primaries it makes sense to think of the primaries as the basis of a linear color space
- Note that the space is infinite dimensional
- The space of valid colors is a 3D-subspace
  - problem: the basis is not necessarily orthogonal



Linear color spaces

- **RGB:** primaries are monochromatic energies are 645.2nm, 526.3nm, 444.4nm.
- color matching functions have negative parts
- some colors can be matched only subtractively



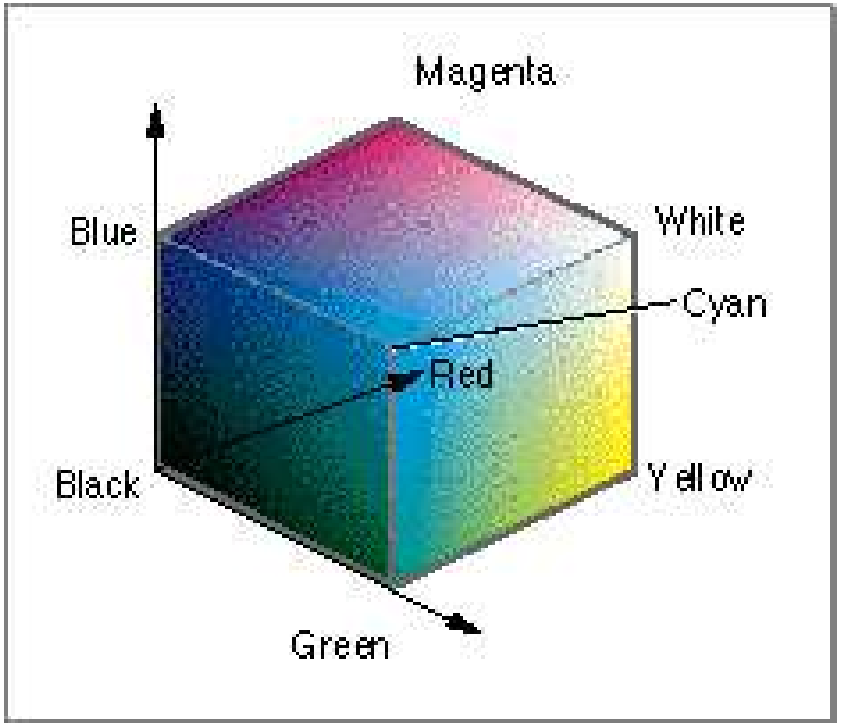


RGB space

- You can think about this color space is a cube

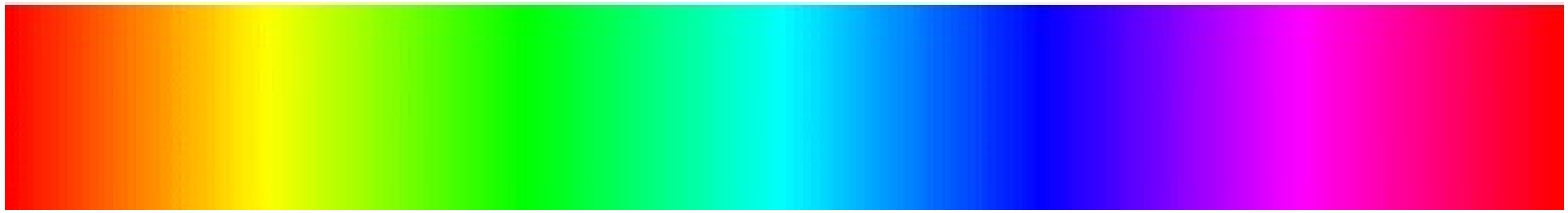
- axes represent the amount of red, green, and blue

- each color has coordinates between 0 and 1 (0 and 255)



Color perception

- human perception of color is best described in terms of three fundamental color properties
- hue: this is what you would refer to as the color itself

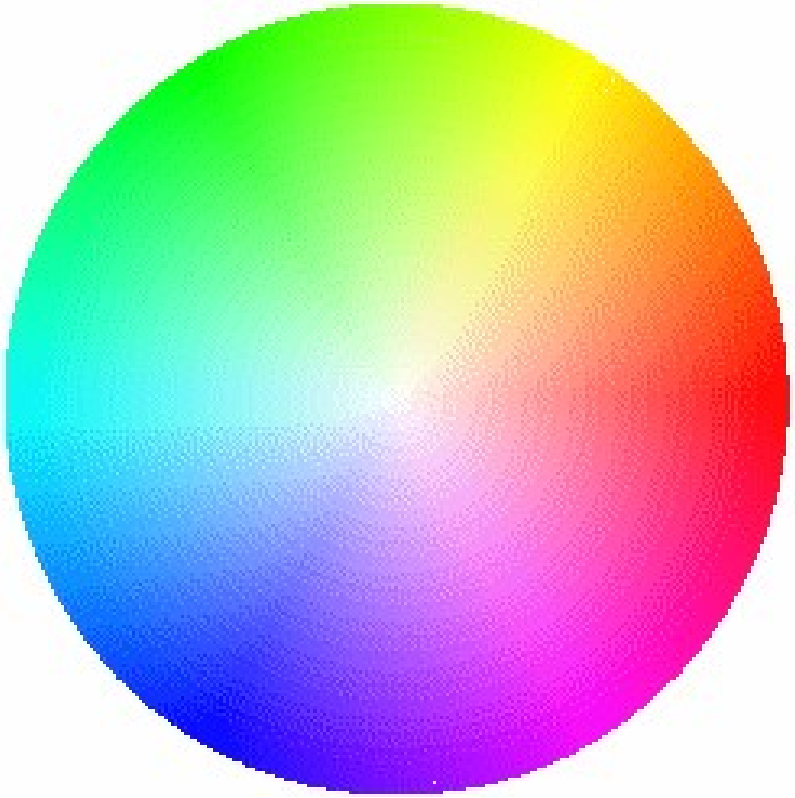


e.g. red vs yellow

- note that there is a circular structure (we start and finish with red)
- for this reason color is usually visualized in a color wheel

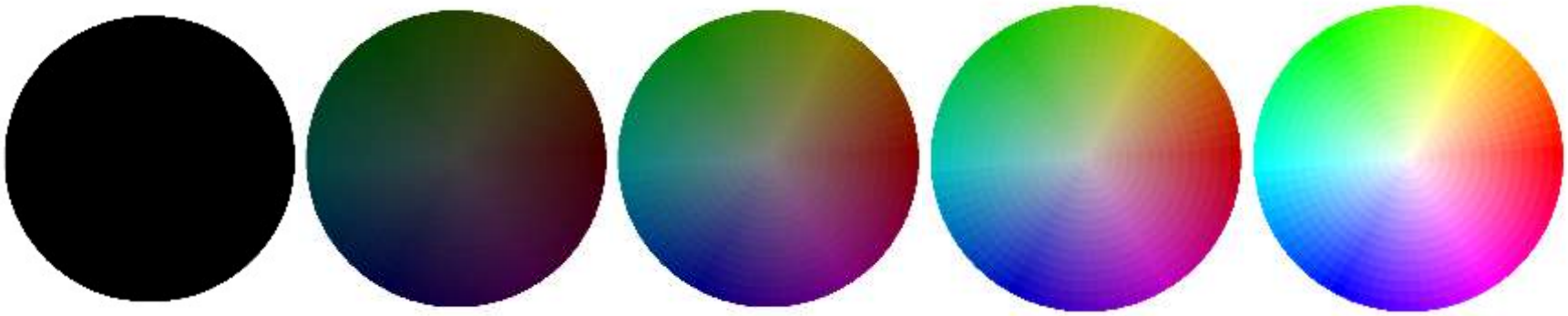
Color perception

- saturation: this is the adjective: e.g. “vivid” red, “pale” yellow
- it is the radial coordinate on the color wheel
- colors at the center are unsaturated, colors at the boundaries are highly saturated



Color perception

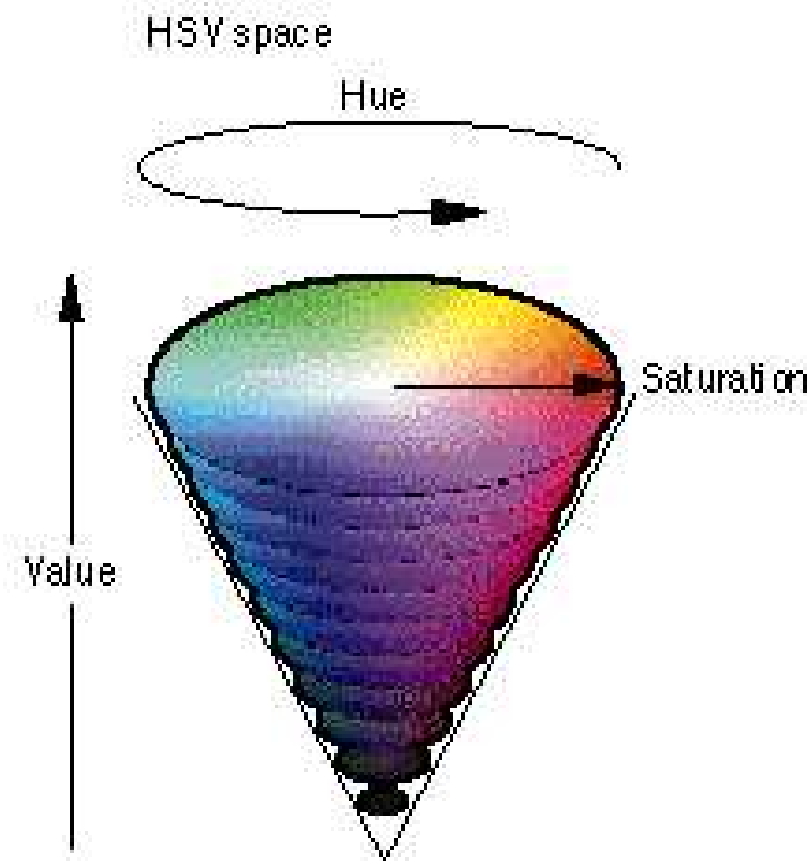
- intensity: is the amount of brightness
- “dark” yellow vs “light” yellow
- the color wheel can be replicated at each intensity level



- Note that:
  - the center is always the intensity level
  - at zero intensity, hue and saturation are irrelevant
  - saturation becomes more important at higher intensities

Perceptual color space: HSV

- **HSV:** hue, saturation, value (intensity) is a natural "perceptual" color space
- very non-linear, colors form a cone
  - this can be bad for some applications
  - but good for others



Linear Filtering

- smoothing is implemented with linear filters
- given an image $x(n_1,n_2)$, filtering is the process of convolving it with a kernel $h(n_1,n_2)$

$$y(n_1,n_2) = \sum_{k_1 k_2} x(k_1,k_2) h(n_1 - k_1, n_2 - k_2)$$

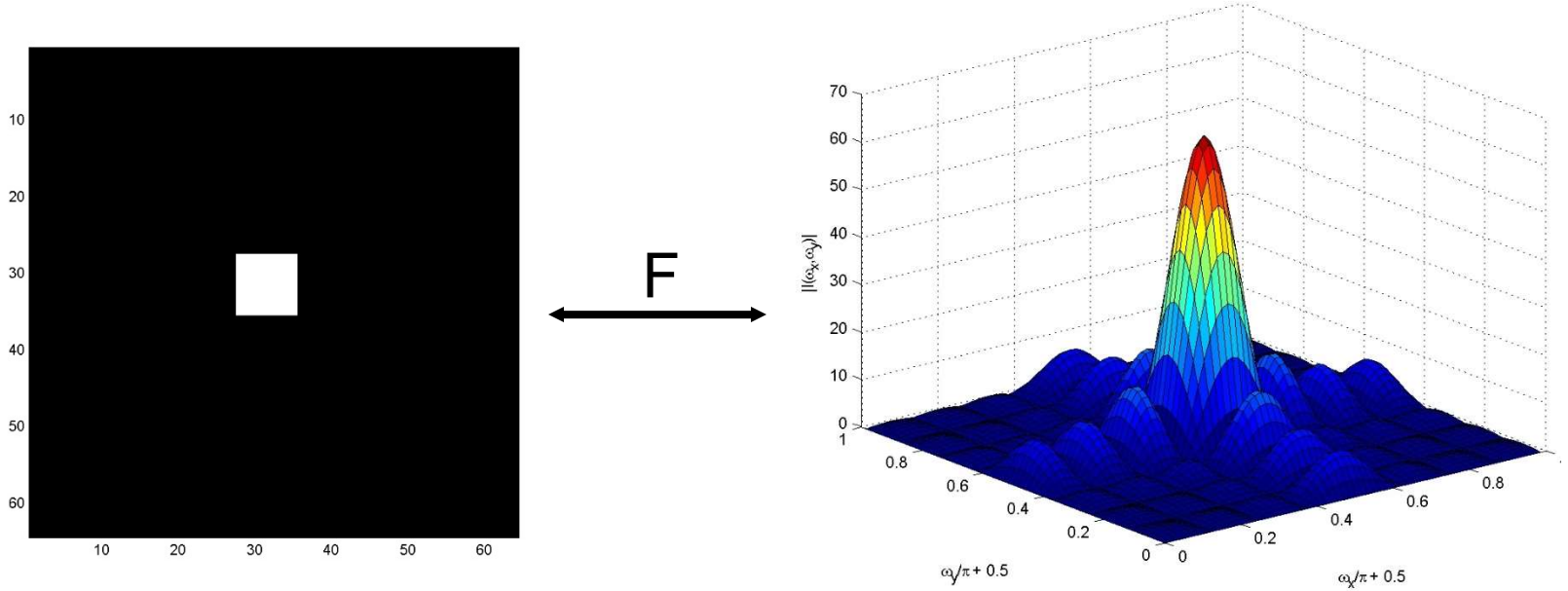
- some very common operations in image processing are nothing but filtering, e.g.
  - smoothing an image by low-pass filtering
  - contrast enhancement by high pass filtering
  - finding image derivatives
  - noise reduction

Popular filters

- box function

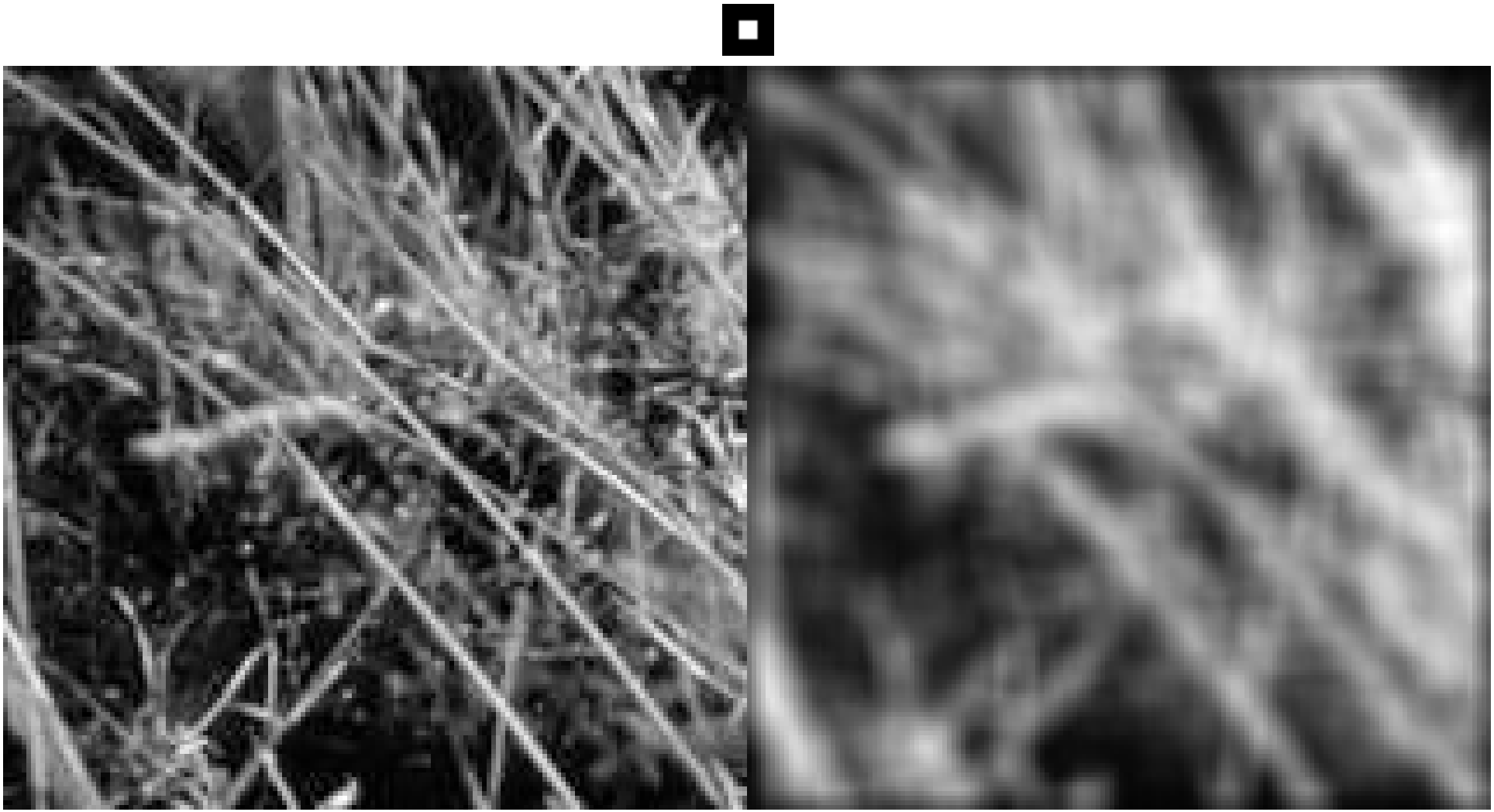
$$R_{N_1 x N_2}(n_1, n_2) = \begin{cases} 1, & 0 \le n_1 \le N_1 - 1, 0 \le n_2 \le N_2 - 1 \\ 0 & otherwise \end{cases}$$

- Fourier transform of a box is the sinc, low-pass filter



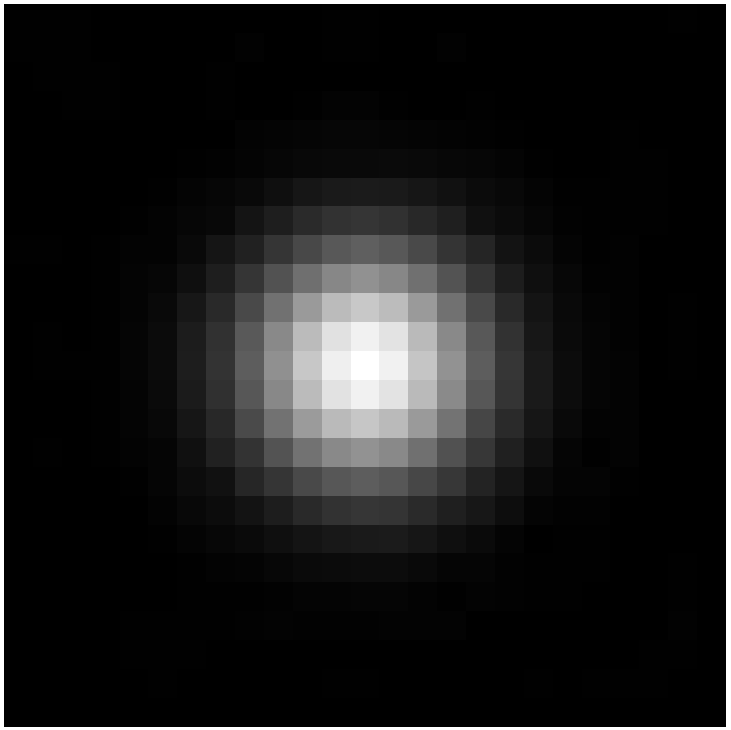
- side-lobes produce artifacts, smoothed image does not look like the result of defocusing

Example: Smoothing by Averaging



Camera defocusing

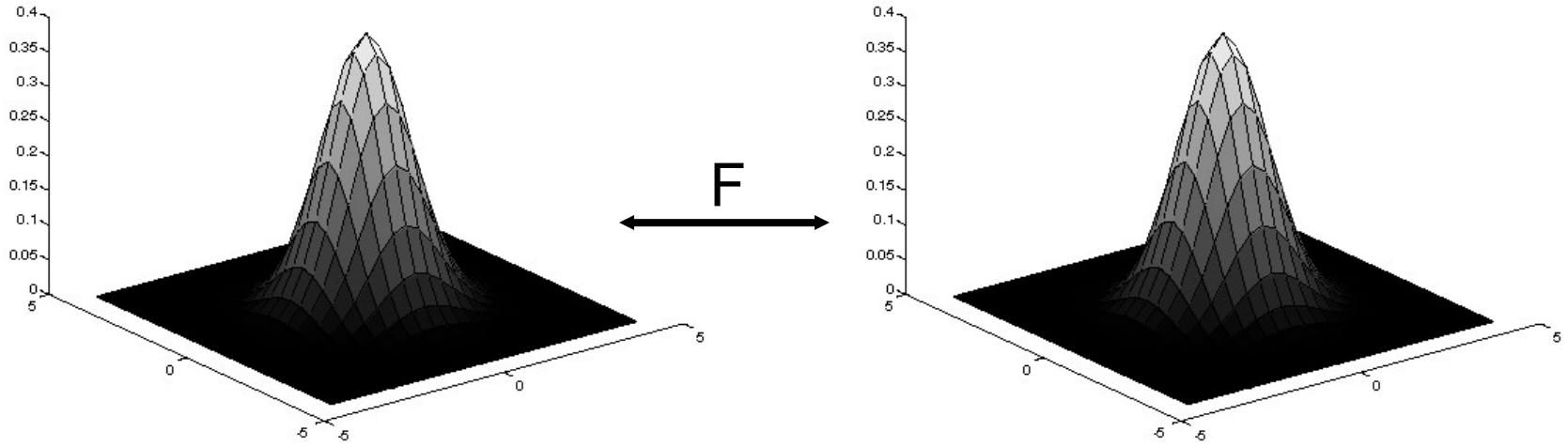
- if you point an out-of-focus camera at a very small white light (e.g. a light-bulb) at night, you get something like this
- the light can be thought of as an impulse
- this must be the impulse response
- well approximated by a Gaussian
- this is a more natural filter for image blur than the box



$$h(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

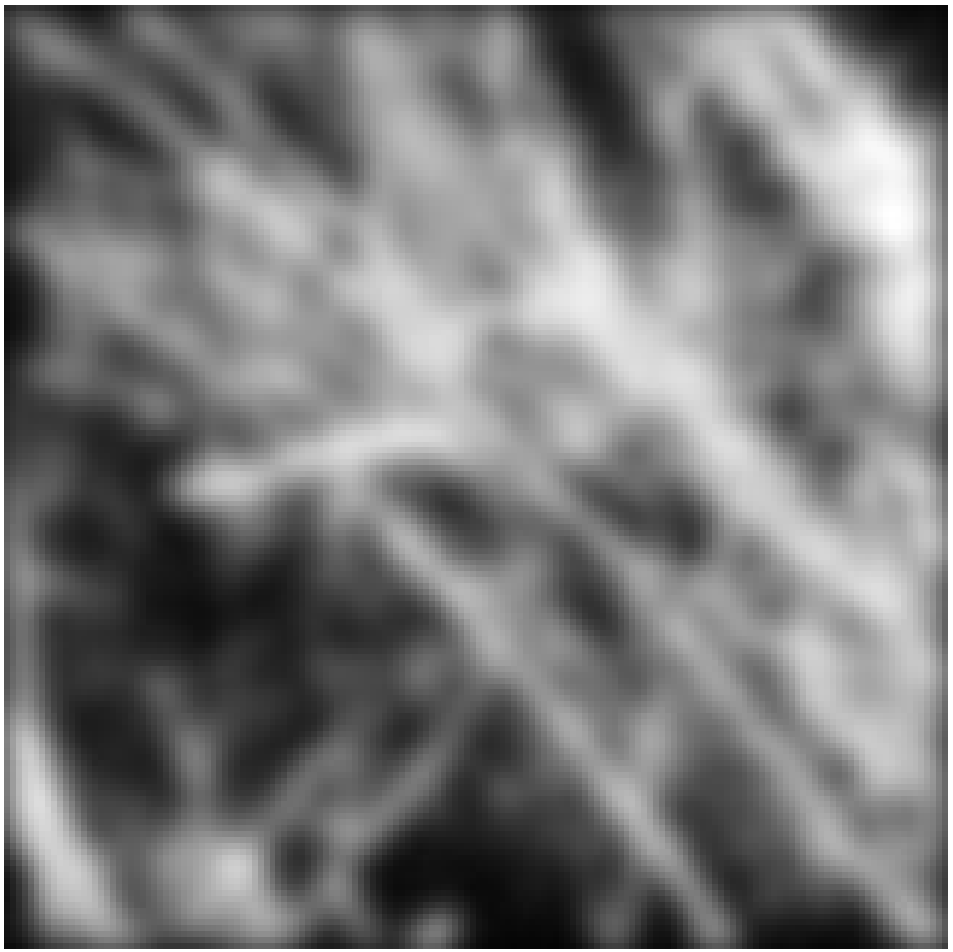
The Gaussian filter

- the Fourier transform of a Gaussian is a Gaussian
  (σ_x,σ_y) / $(1/\sigma_{w1},1/\sigma_{w2})$

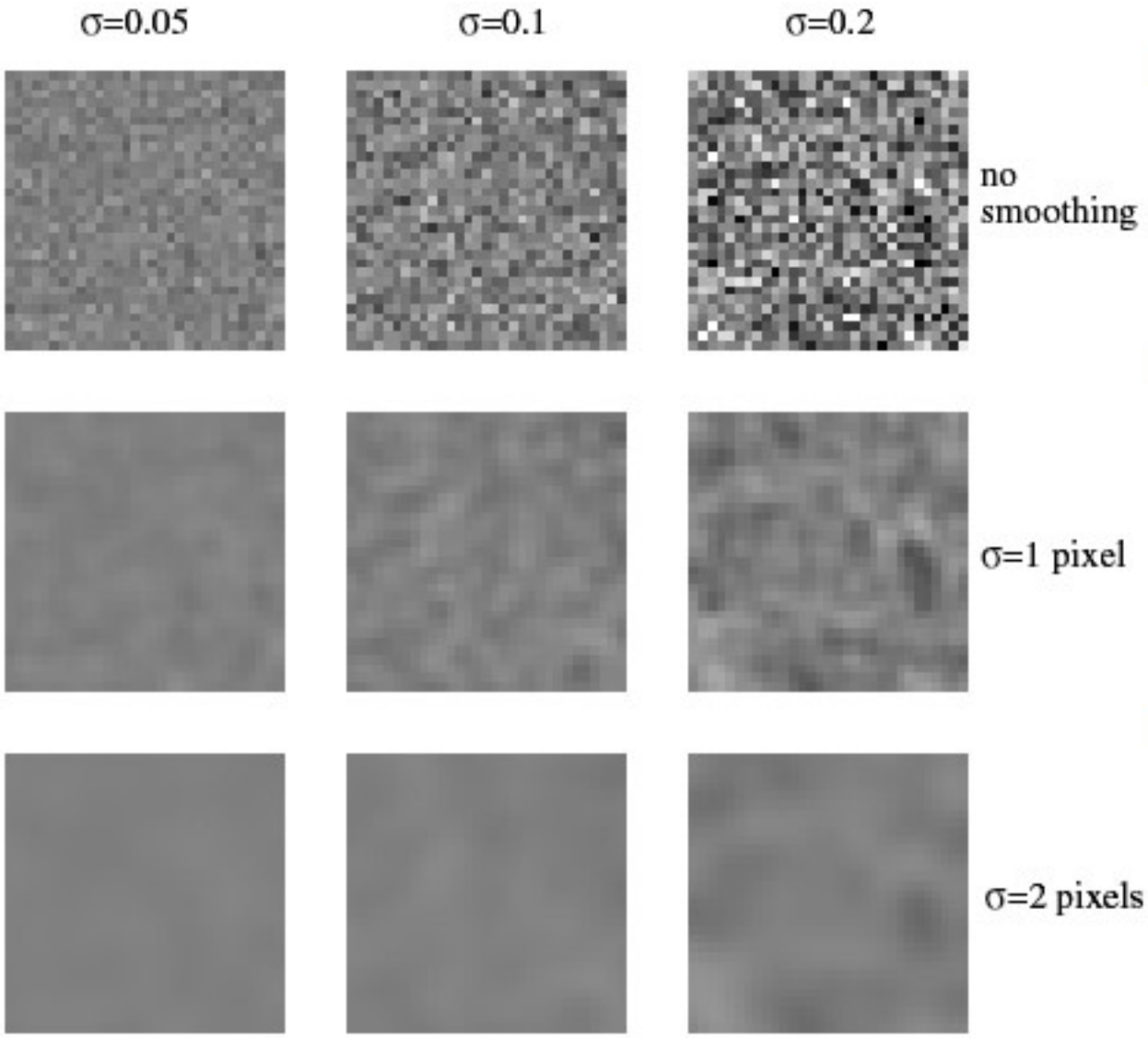


- note that there are no annoying side-lobes

Smoothing with a Gaussian



Role of the variance



- the variance controls the amount of smoothing
- each column shows different realizations of an image of gaussian noise
- each row shows smoothing with gaussians of different σ

Gradients

- for a 2D function, *f(x,y)* the gradient at a point (x_0,y_0)

$$\nabla f(x_0,y_0) = \left(\frac{\partial f}{\partial x}(x_0,y_0), \frac{\partial f}{\partial y}(x_0,y_0)\right)^T$$
$$= \left(f_x(x_0,y_0), f_y(x_0,y_0)\right)^T$$

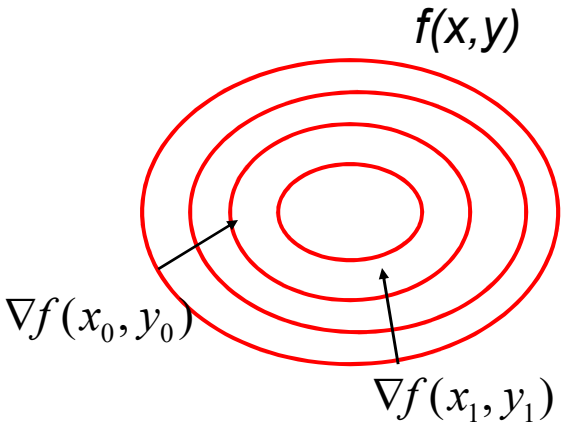
  is the direction of greatest increase at that point

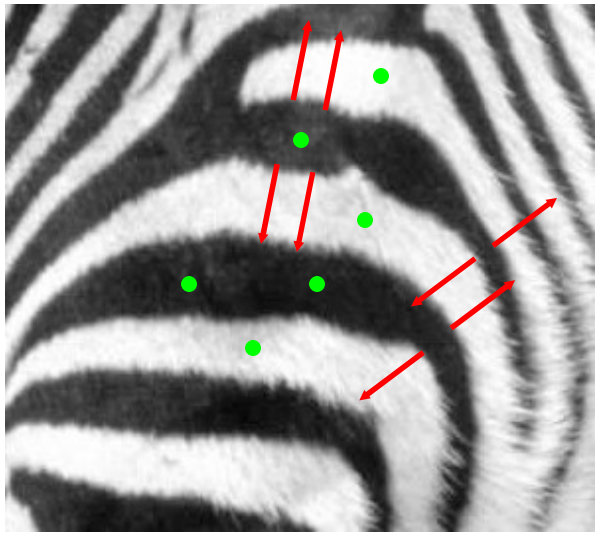
- the gradient magnitude

$$\left\|\nabla f(x_0,y_0)\right\|^2 = \left(\frac{\partial f}{\partial x}(x_0,y_0)\right)^2 + \left(\frac{\partial f}{\partial y}(x_0,y_0)\right)^2$$

  measures the rate of change

- it is large at edges!



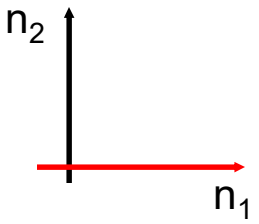


– large gradient magnitude
– small gradient magnitude

Finite difference kernels

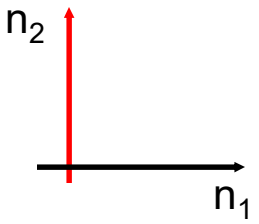
- in two dimensions we have various possible kernels
- e.g. , $N_1=2$, $N_2=3$, derivative along n_1, *(line $n_2=k$)* (horizontal)

$$\begin{matrix} 0 & 0 \\ 1 & -1 \\ 0 & 0 \end{matrix} \qquad \begin{matrix} 1 & -1 \\ 1 & -1 \\ 1 & -1 \end{matrix}$$



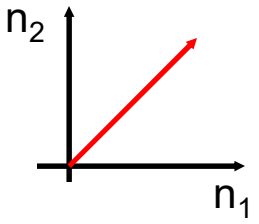
- derivative along n_2, *(line $n_1=k$)* (vertical)

$$\begin{matrix} 0 & -1 & 0 \\ 0 & 1 & 0 \end{matrix} \qquad \begin{matrix} -1 & -1 & -1 \\ 1 & 1 & 1 \end{matrix}$$



- derivative along line $n_1=n_2$ (diagonal)

$$\begin{matrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{matrix} \qquad \begin{matrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{matrix}$$



Finite differences

- Which one do we have here?

Finite differences and noise

- because they perform high-pass filtering, finite difference filters respond strongly to noise
- generally, the larger the noise the stronger the response
- for noisy images it is usually best to apply some smoothing before computing derivatives
- what do mean by noise?
- we only consider the simplest model
  - independent stationary additive Gaussian noise
  - the noise value at each pixel is given by an independent draw from the same normal probability distribution

$$y(n_1, n_2) = x(n_1, n_2) + \varepsilon(n_1, n_2), \qquad \varepsilon \sim N(0, \sigma^2)$$

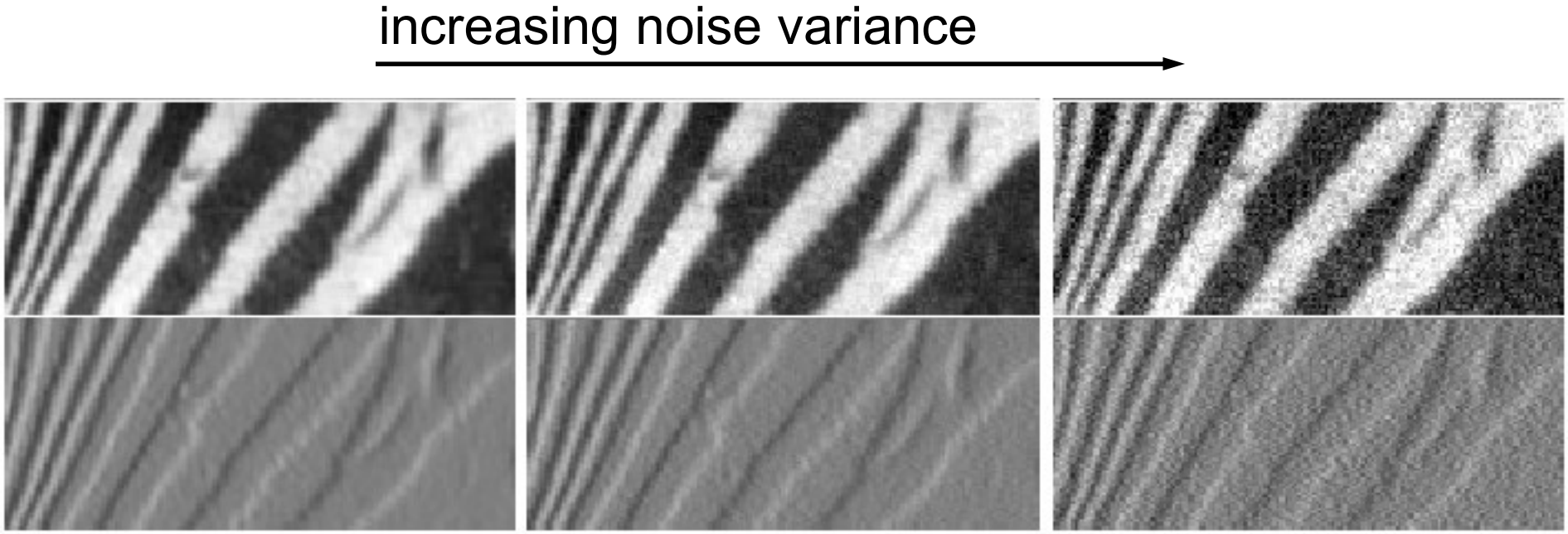
Noise levels

sigma=1
(noise)

Noise levels

sigma=16
(noise)

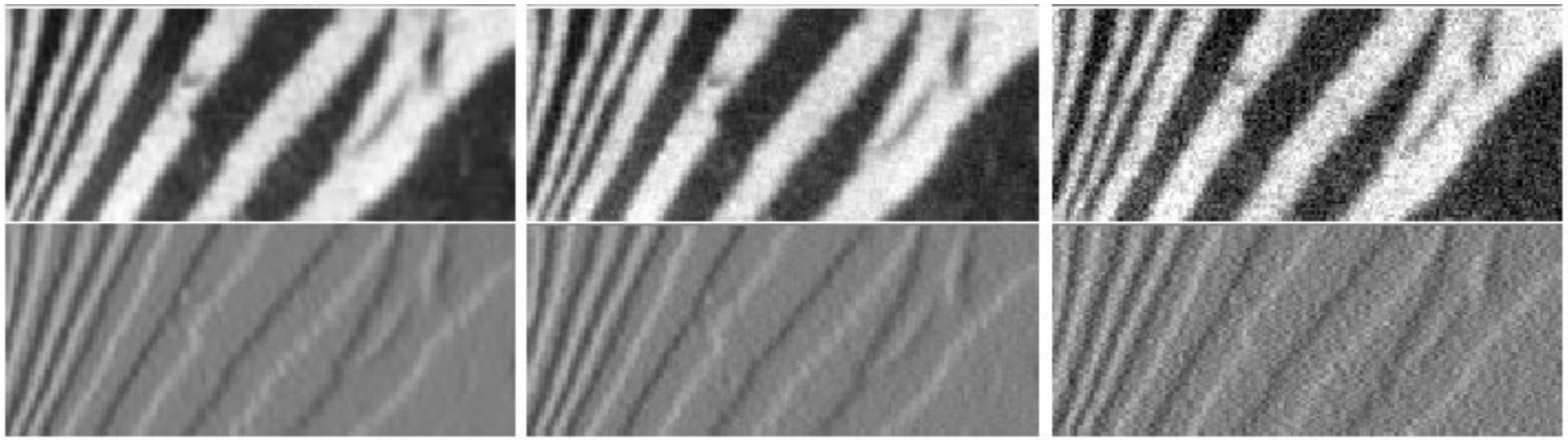
Finite differences responding to noise



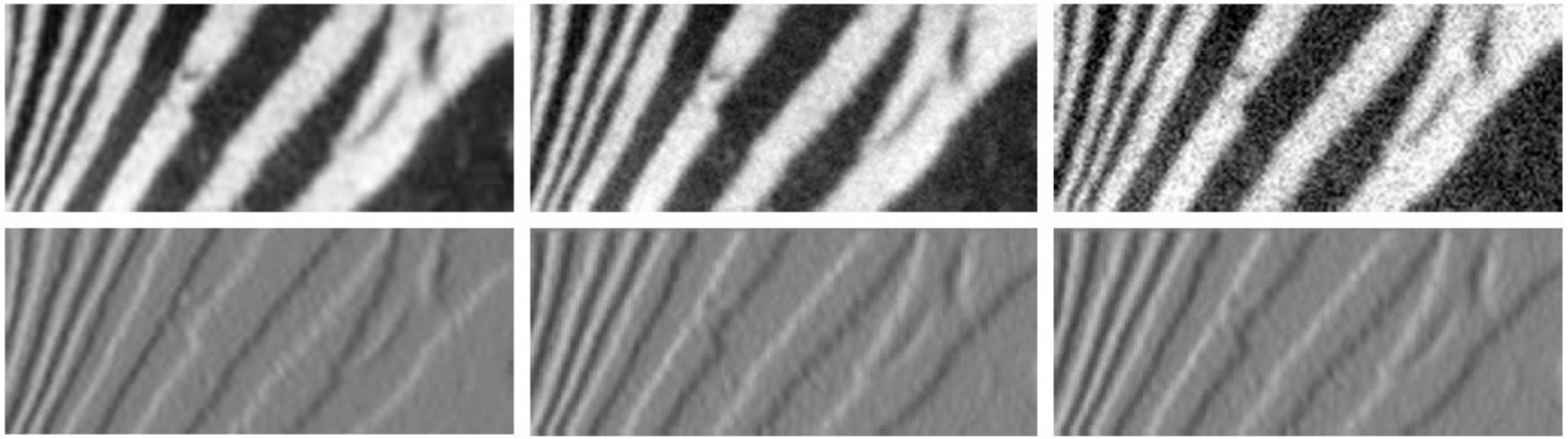
- note that as the noise variance increases the estimates of the image derivative are also very noisy
- Would a larger filter do better?

Smoothed derivatives

no smoothing



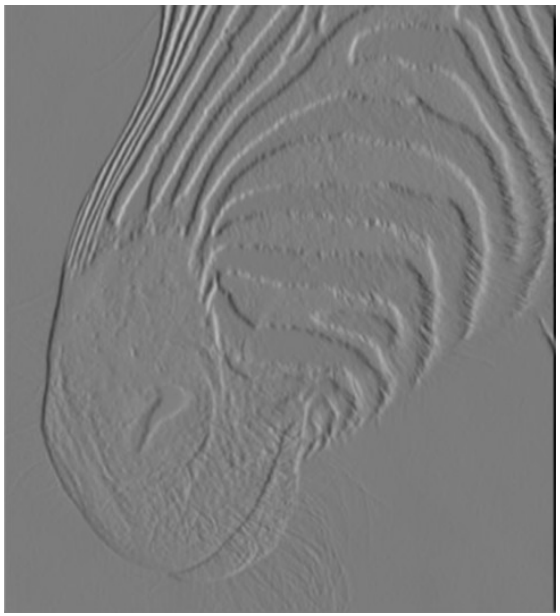
with smoothing



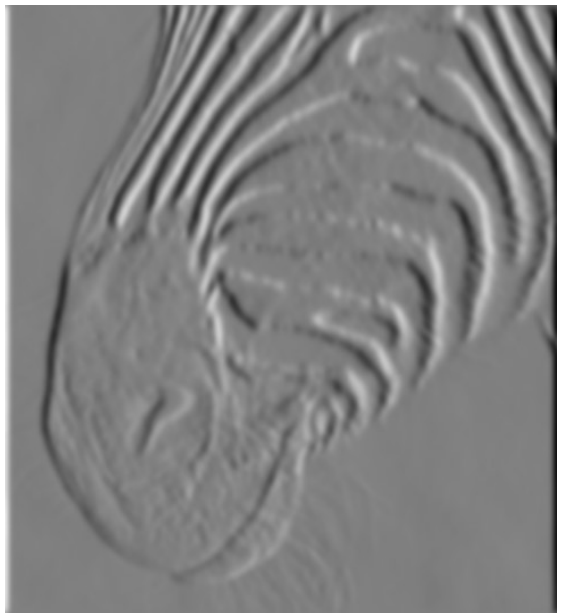
Choosing the right scale

- the scale of the smoothing filter affects
  - derivative estimates,
  - the semantics of the derivative image
- trade-off between noise and ability to detect detail

1 pixel



3 pixels



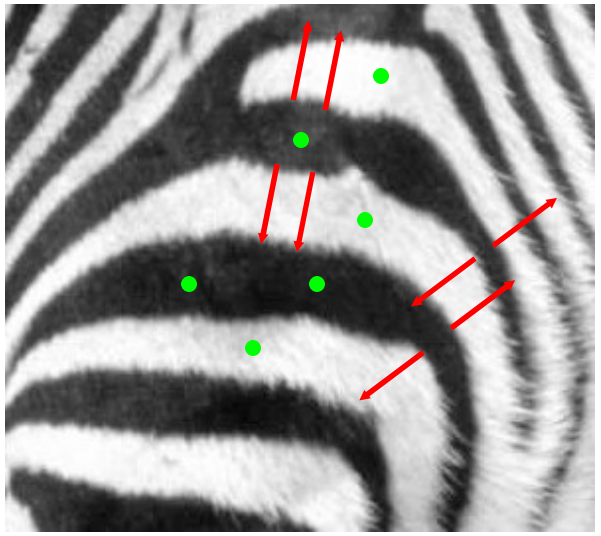
7 pixels

Gradients and edges

- by now we have a good idea of how to differentiate images
- remember that edges are points of large gradient magnitude
- edge detection strategy
  1. determine magnitude of image gradient

$$\|\nabla f(x_0, y_0)\|^2 = \left(\frac{\partial f}{\partial x}(x_0, y_0)\right)^2 + \left(\frac{\partial f}{\partial y}(x_0, y_0)\right)^2$$

  2. mark points where gradient magnitude is particularly large wrt neighbours (ideally, curves of such points)



– large gradient magnitude
– small gradient magnitude

Looks easy but

- 1) gradient magnitude at different scales is different (see below); which should we choose?
- 2) gradient magnitude is large along thick trail; what are the significant points?
- 3) how do we link the relevant points up into curves?