COMPUTER VISION 2018 - 2019

>IMAGE FORMATION

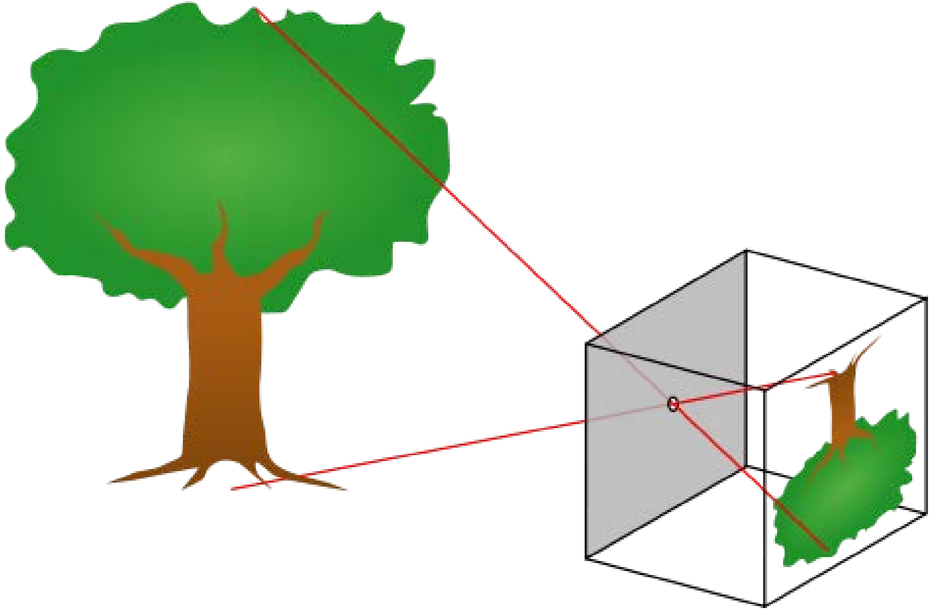
UTRECHT UNIVERSITY

RONALD POPPE

CAMERAS

2D projection of 3D world onto an image:

- “shape”: geometry
- “color”: radiometry



OUTLINE

Camera geometry

Camera radiometry

Issues in image formation

Color spaces and distances

Assignment

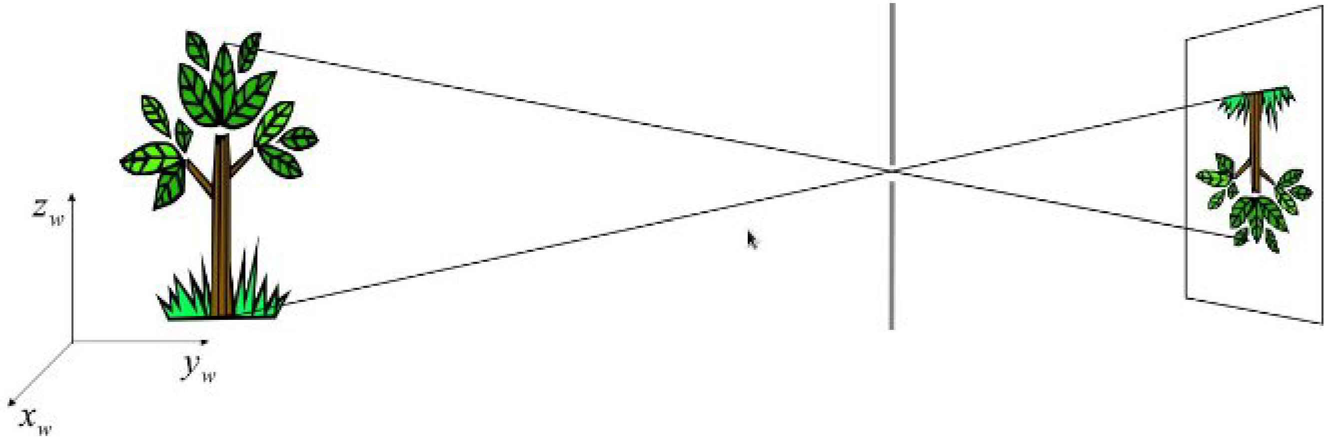
CAMERA GEOMETRY

CONCEPT

We have two coordinate systems:

- 3D world coordinates (in meters)
- 2D image coordinates (in pixels)

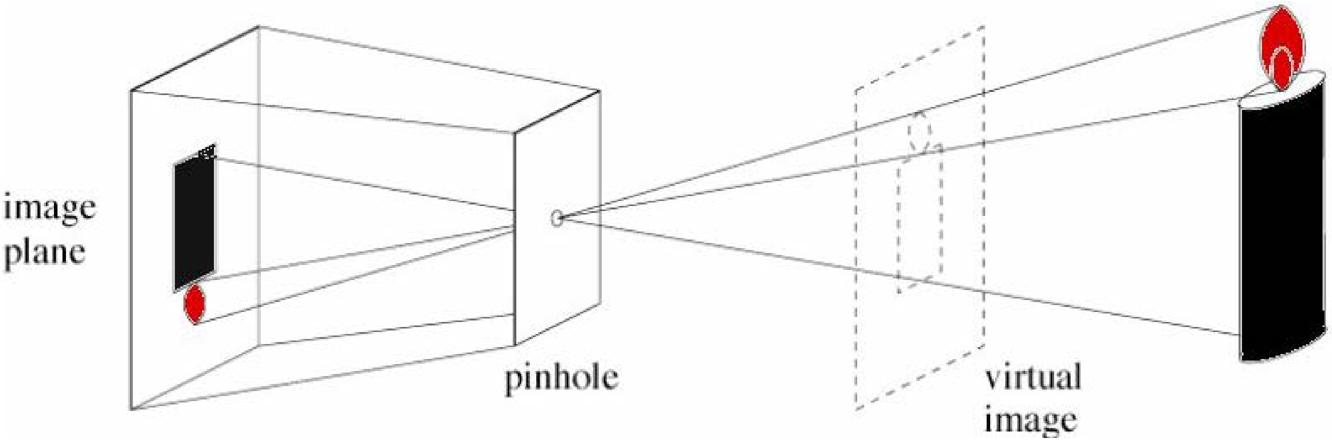
How to model the projection from 3D to 2D?



PINHOLE CAMERA MODEL

Often, we assume a pinhole camera model

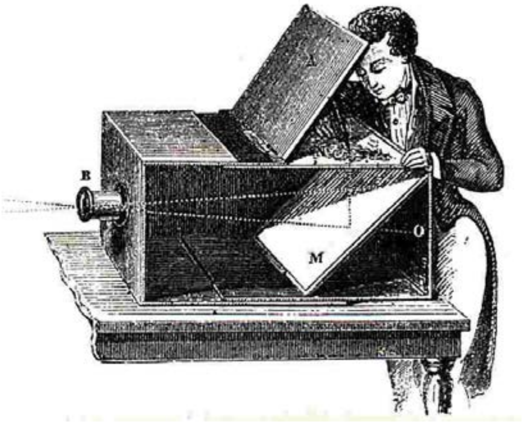
- Think of light as rays
- The “lens” is a point
- The presence of a virtual image can be assumed

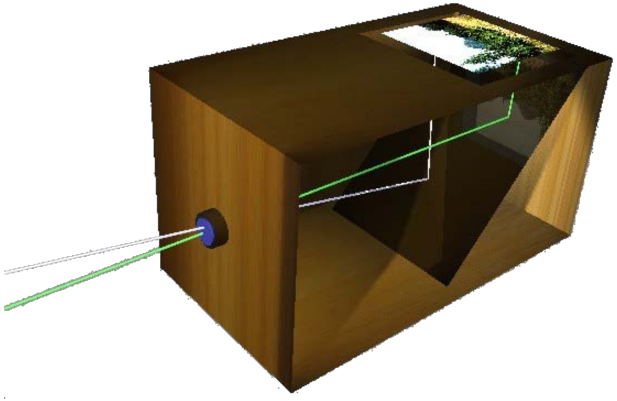


PINHOLE CAMERA MODEL[2]

Camera obscura can be considered a pinhole camera

- Light-receptive paper used to “record” image
- Drawback: limited amount of light will pass through the hole
- Long exposure times are needed for a good “image”

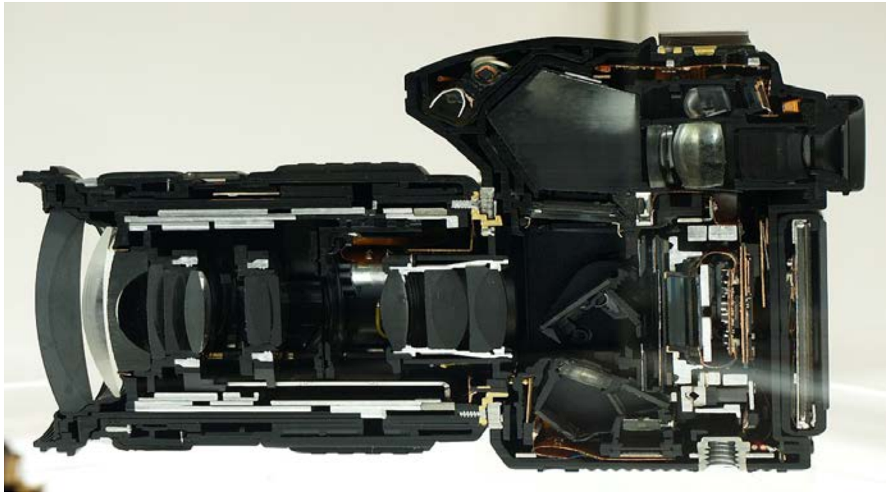


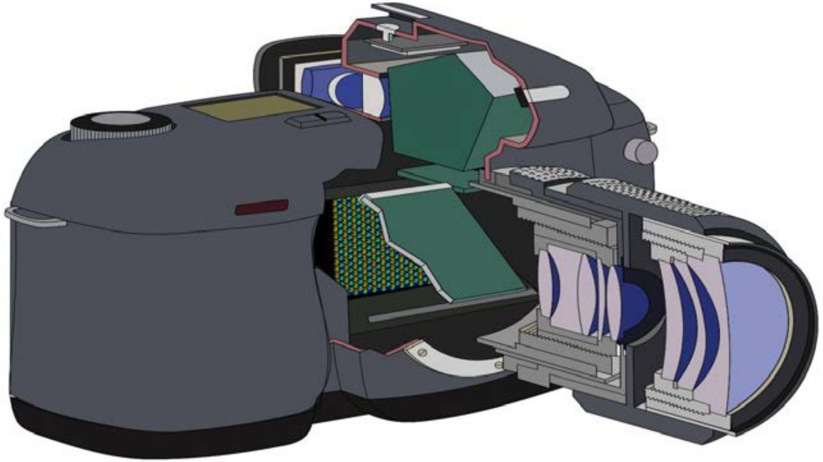


DIGITAL CAMERAS

Modern (digital) cameras use lenses to focus light

- Often multiple lenses

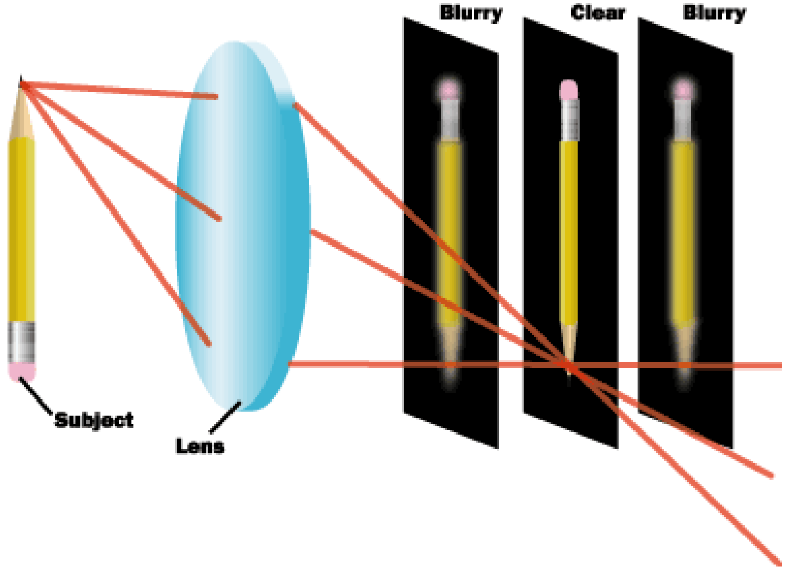




DIGITAL CAMERAS[2]

Focusing of light has advantages:

- Larger lens area so shorter exposure times
- Zooming possible, so focus at different distances
- “Moves the virtual image”



FROM 3D TO 2D

How a 3D scene/object appears on the 2D image depends on:

- The properties of the lens
- The object/scene itself

In an image projection, the depth information is lost

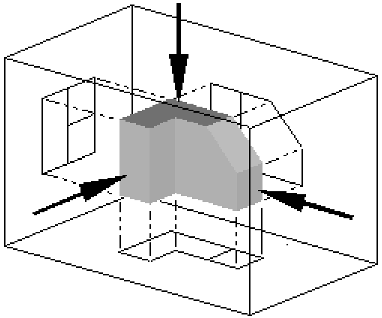
How depends on the assumptions of the projection:

- Orthographic projection
- Perspective projection

ORTHOGRAPHIC PROJECTION

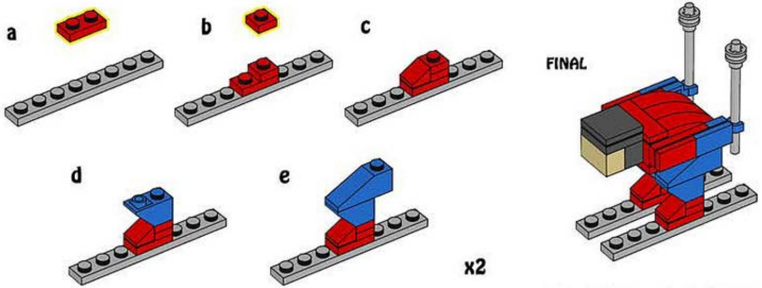
Easiest way is to drop the depth information (z-dimension):

$$\boldsymbol{x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \boldsymbol{p}$$



With x a 2D coordinate and p a 3D point.

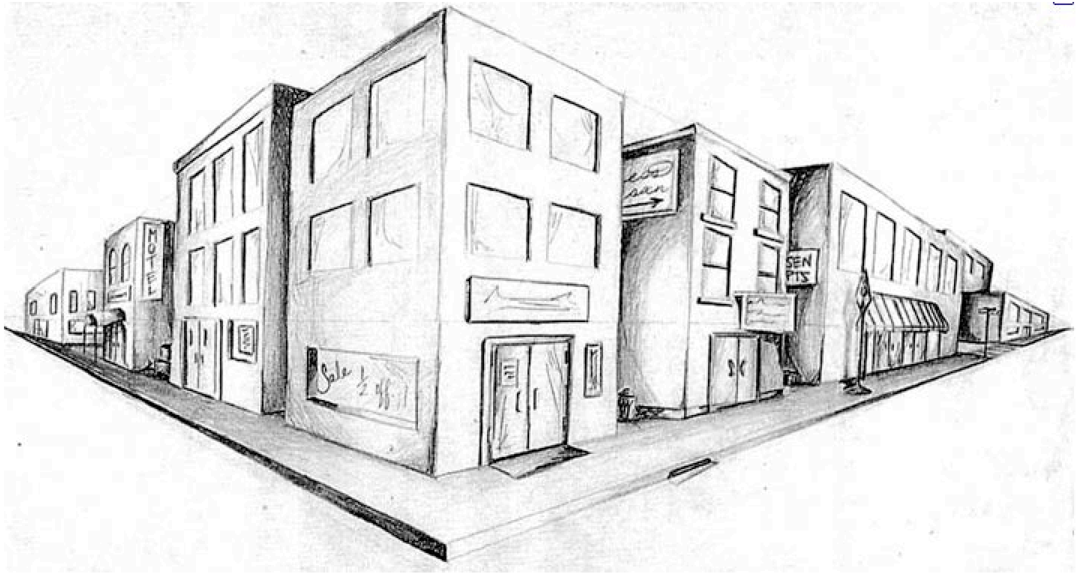
Not consistent with how we see the world



PERSPECTIVE PROJECTION

More realistic: objects that are further away appear smaller:

- Depth affects projection
- Straight lines are still straight after projection



CAMERA MATRIX

A camera matrix *P* represents the projection from a 3D point in the world to a 2D point in the image.

- Images have a 2D local axis system (pixels)
- The world has some arbitrary 3D origin and 3D axis system (in meters)

***P* contains extrinsic and intrinsic parameters:**

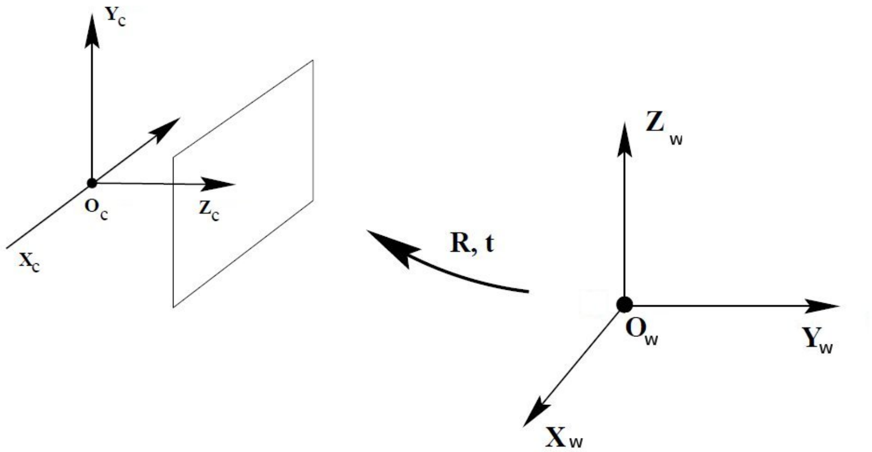
- Extrinsic: convert 3D world to 3D camera coordinates
- Intrinsic: convert 3D camera to 2D image coordinates

We will first discuss the extrinsic parameters, then the intrinsic ones

EXTRINSIC PARAMETERS

Express 3D world coordinates in 3D camera coordinates:

- Account for the position and orientation of the camera in the world
- Recall a 3D rotation: $\bar{\boldsymbol{x}}' = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \bar{\boldsymbol{x}}$
- We have to determine extrinsic parameters ***R*** (3x3) and ***t*** (3x1)



EXTRINSIC PARAMETERS[2]

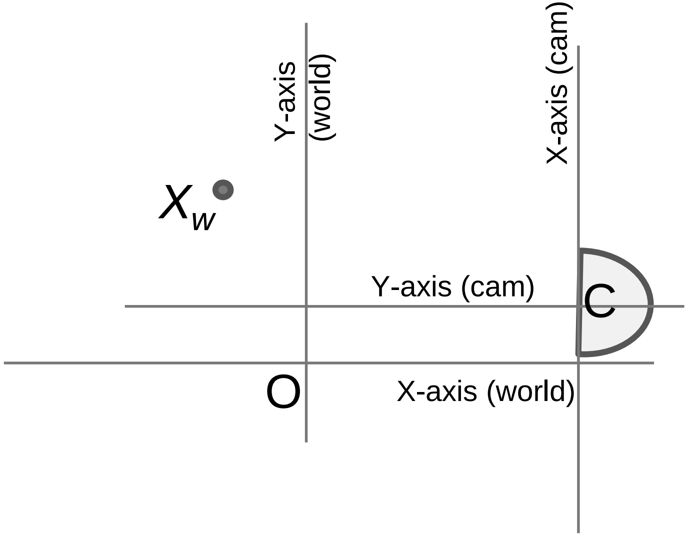
What we do is:

- Find the difference in orientation between camera axis system and world axis system: ***R***
- Find the location of world origin ***O*** seen from camera center ***C***: ***t***

EXTRINSIC PARAMETERS[3]

Example in 2D:

- For convenience: camera ***C*** direction is orthogonal to world
- We "look" along the camera's y-axis (usually z-axis)



$\boldsymbol{X_w}$ is our world coordinate (-1, 3)
Our camera center ***C*** is at (4,1)

It's rotated 90 degrees counter clockwise:

- Cos(-90) = 0, Sin(-90) = -1
- So $\boldsymbol{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$

Origin ***O*** expressed in ***C***'s axis system: (-1, 4):

$$\overline{\boldsymbol{X_c}} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \overline{\boldsymbol{X_w}} = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix}$$

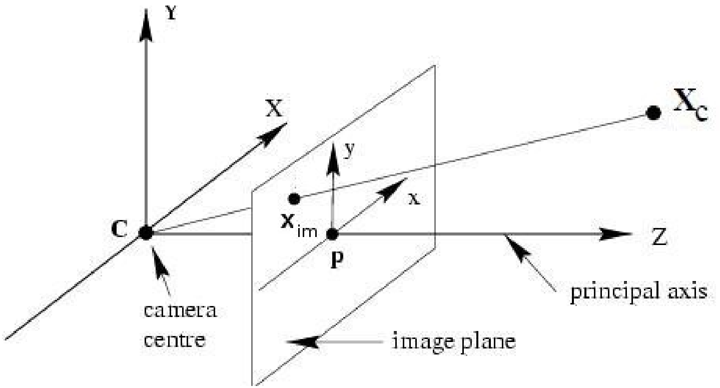
INTRINSIC PARAMETERS

We now project the point X_C in 3D camera coordinates on the 2D image plane (x_{im})

- $\boldsymbol{X_C}$ is in camera coordinates, with origin **C** and principal axis *Z*

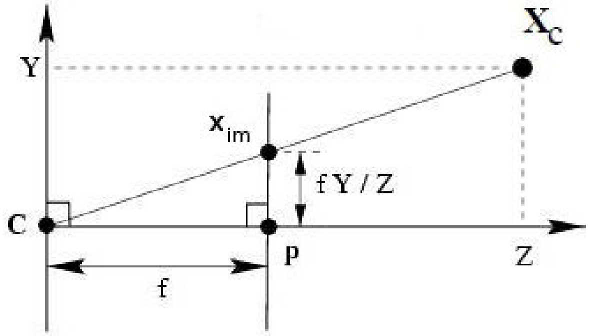
Matrix K represents the projective transformation of a point in 3D camera coordinates to a 2D image point:

$$\boldsymbol{x_{im}} = \begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = \boldsymbol{K} \begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix}$$



INTRINSIC PARAMETERS[2]

In 2D (side view) this looks like this:



***f* is focal length, or distance to (virtual) image plane**

Our aim is to find x_{im} (x_{im}, y_{im}, f), given X_C and f:

$$\begin{bmatrix} x_{im} \\ y_{im} \\ f \end{bmatrix} = \frac{f}{Z_c} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Note: $(x_{im,} y_{im})^T \neq (X_C, Y_C)^T$ **but they represent the same point**

INTRINSIC PARAMETERS[3]

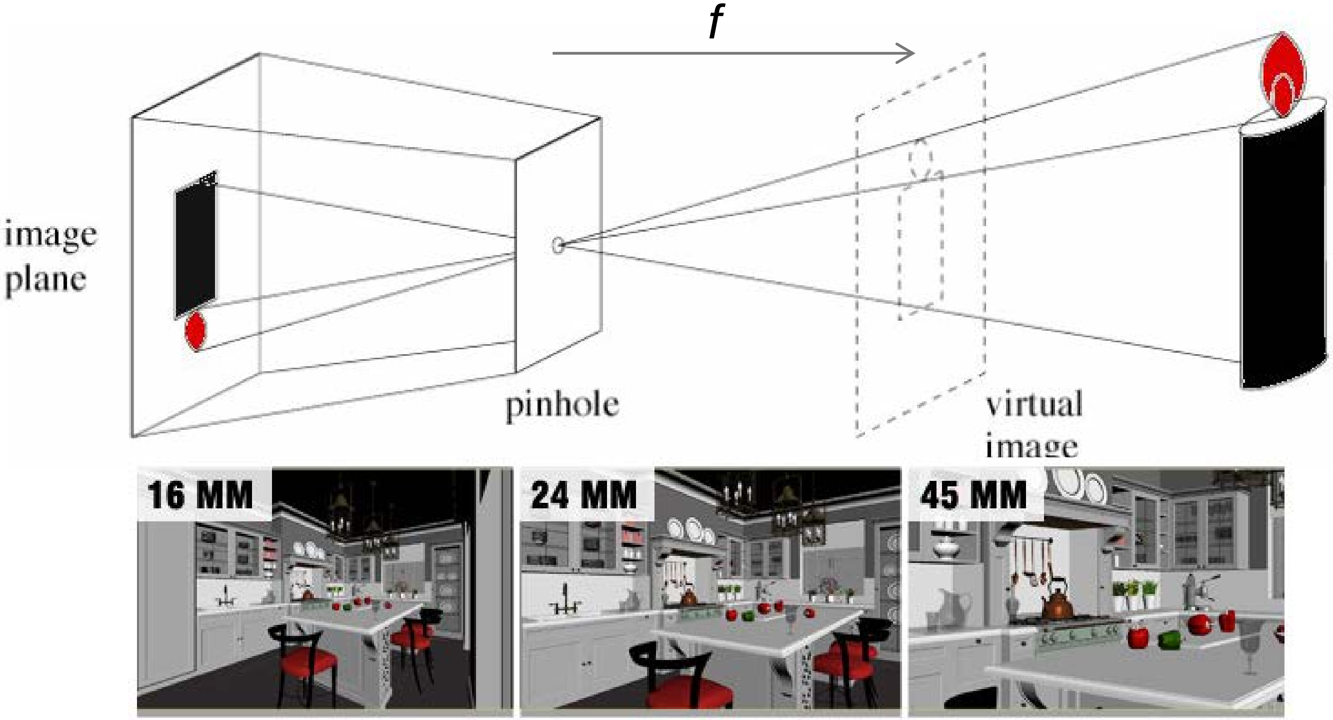
Instead of vector notation: $\begin{bmatrix} x_{im} \\ y_{im} \\ f \end{bmatrix} = \frac{f}{Z_c} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$

We can also use matrix notation: $\begin{bmatrix} x_{im} \\ y_{im} \\ f \end{bmatrix} = \begin{bmatrix} f/Zc & 0 & 0 \\ 0 & f/Zc & 0 \\ 0 & 0 & f/Zc \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$

As x_{im} is typically in pixels, so is *f*.

INTRINSIC PARAMETERS[4]

Effect of increasing and decreasing *f*

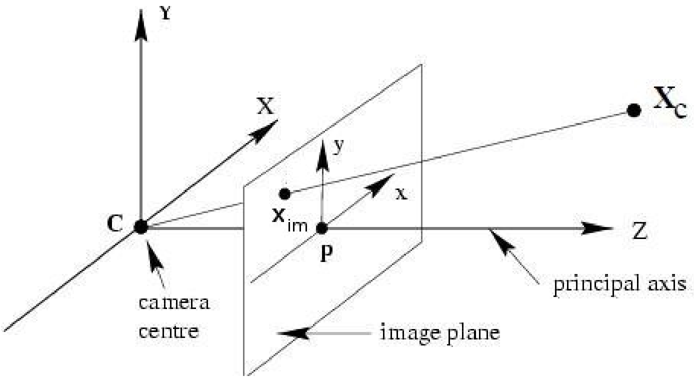


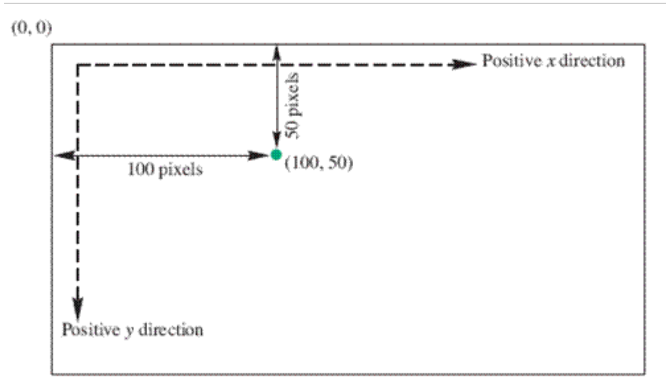
INTRINSIC PARAMETERS[5]

We can now project 3D to 2D, but the 2D origin is in the center:

- Pixels typically numbered from left-upper corner to right-lower corner
- We need a translation

Not possible in matrix-form in current notation!





HOMOGENEOUS COORDINATES

The x representation is termed inhomogeneous

Homogeneous coordinate $\tilde{\boldsymbol{x}}$ is an alternative representation:

$\tilde{\boldsymbol{x}} = (\tilde{x} \quad \tilde{y} \quad \tilde{z} \quad \tilde{w}) = (\tilde{w}x \quad \tilde{w}y \quad \tilde{w}z \quad \tilde{w}) = \tilde{w}(x \quad y \quad z \quad 1)$ **or,**

$\boldsymbol{x} = (x \quad y \quad z) = (\tilde{x}/\tilde{w} \quad \tilde{y}/\tilde{w} \quad \tilde{z}/\tilde{w})$**, and**

$\bar{\boldsymbol{x}} = (x \quad y \quad z \quad 1) = (\tilde{x}/\tilde{w} \quad \tilde{y}/\tilde{w} \quad \tilde{z}/\tilde{w} \quad \tilde{w}/\tilde{w})$**.**

When $\tilde{w} = 0$, the point is at infinity.

Conversion is straightforward for $\tilde{w} = 1$.

HOMOGENEOUS COORDINATES[2]

Why homogeneous coordinates?

- Points at infinity can be expressed
- Transformations can be expressed in matrix form
- Perspective projections are straightforward

Translation in homogeneous coordinates becomes multiplication:

$$\tilde{\boldsymbol{x}}' = \begin{bmatrix} \tilde{x} + a \\ \tilde{y} + b \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix}$$

INTRINSIC PARAMETERS[6]

Recall: $\boldsymbol{x'}_{\boldsymbol{im}} = \begin{bmatrix} x_{im} \\ y_{im} \\ f \end{bmatrix} = \begin{bmatrix} f/Zc & 0 & 0 \\ 0 & f/Zc & 0 \\ 0 & 0 & f/Zc \end{bmatrix} \boldsymbol{X_c}$

The projection in homogeneous coordinates: $\widetilde{\boldsymbol{x}}'_{\boldsymbol{im}} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \widetilde{\boldsymbol{X}}_{\boldsymbol{C}}$

New $\widetilde{w}$ value is 0, so depth information is lost

We easily add translation (x_0, y_0) in the image plane: $\widetilde{\boldsymbol{x}}'_{\boldsymbol{im}} = \begin{bmatrix} f & 0 & x_0 & 0 \\ 0 & f & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \widetilde{\boldsymbol{X}}_{\boldsymbol{C}}$

INTRINSIC PARAMETERS[7]

$\begin{bmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix}$ **is the intrinsic camera matrix *K***

- It projects and translates
- Does not take into account properties of the lens or sensor apart from the focal length

INTRINSIC PARAMETERS[8]

Typically, we use a generalized version of *K*: $\begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$

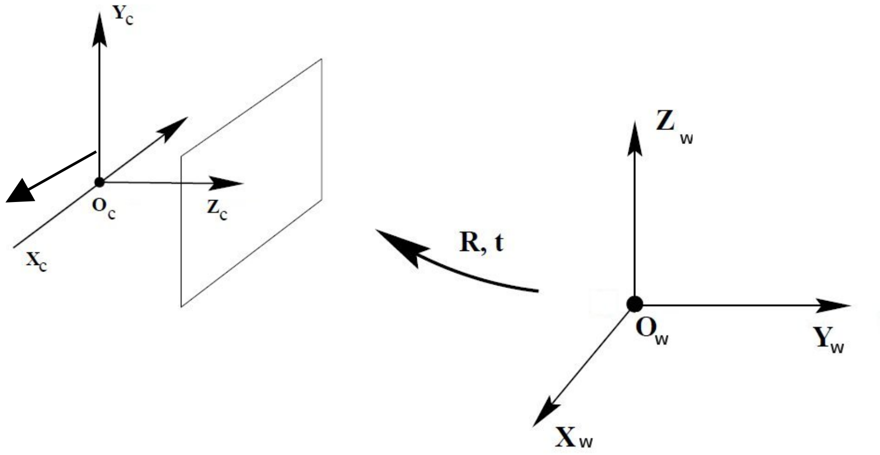
$s = \tan\phi$ **is the skew coefficient, with ϕ the angle between the image axes**

- If $s=0$ and $f_x=f_y$, then pixels are perfectly square
- If $s=0$ and $f_x \neq f_y$, then pixels are rectangular
- If $s \neq 0$ and $f_x=f_y$, then pixels are skewed

CAMERA MATRIX REVISITED

Perspective projection consists of:

- Projection from 3D world to 3D camera coordinates: extrinsic
- Projection from 3D camera to 2D image coordinates: intrinsic



CAMERA MATRIX REVISITED^2

Extrinsic parameters (project from world to 3D camera coordinates):

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = [\boldsymbol{R} \quad \boldsymbol{t}] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Intrinsic parameters (project from 3D camera to image coordinates):

$$\begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = \boldsymbol{K} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

CAMERA MATRIX REVISITED[3]

The role of the camera matrix *P*:

- Combine both
- $\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R} \quad \boldsymbol{t}]$

***K* is a 3 x 3 matrix, $[\boldsymbol{R} \quad \boldsymbol{t}]$ a 3 x 4 matrix**

- ***P*** is a 3 x 4 matrix
- Typically not calculated, but left as $\boldsymbol{K}[\boldsymbol{R} \quad \boldsymbol{t}]$

CAMERA MATRIX REVISITED[4]

The 3 x 4 camera matrix $\boldsymbol{P}$ projects 3D world coordinates onto 2D image coordinates:

$$\bar{\boldsymbol{x}}_{\boldsymbol{im}} = \boldsymbol{P}\bar{\boldsymbol{X}}_{\boldsymbol{world}}$$

$$\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R} \quad \boldsymbol{t}]$$

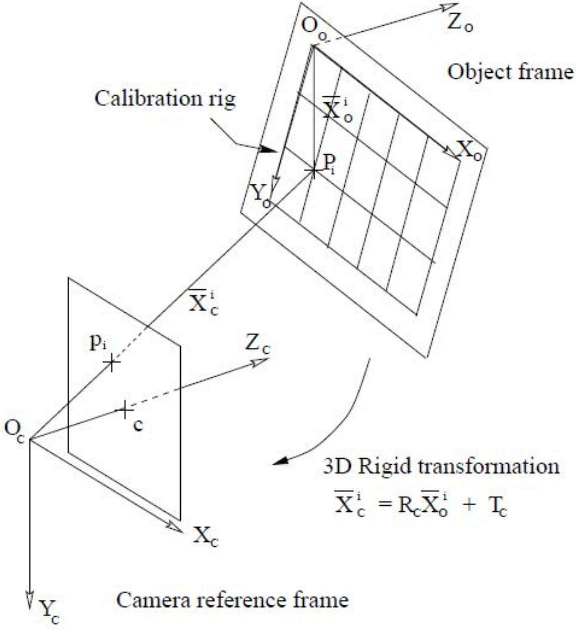
The aim of geometric calibration is to find parameters in *K*, *R* and *t*:

- Calculate ***K*** only once for a fixed camera setting (i.e., zoom)
- Re-calculate ***R*** and ***t*** everytime we move the camera

CALIBRATION

CALIBRATION

Aim of calibration is to determine the extrinsic and intrinsic parameters from images of an object with known properties



CALIBRATION^2

Typically, a chessboard or checkerboard is used:

- Difference between dark and light can be detected robustly
- Distance between crossings is equal
- All crossings are in the same plane

Left upper corner of the first image is taken as the origin

- x- and y-axes along the sides of the checkerboard
- z-axis is orthogonal, so points out of the image.

CALIBRATION[3]

Basic idea:

Set of 3D points

(crossings of the checker-board)

Unknown projection parameters

Set of 2D points

(strong black-white contrasts in 2D directions)

CALIBRATION[4]

The mapping from 3D to 2D is underspecified:

- We need multiple 2D measurements of a 3D scene

Once we know the real distance between crossings, we obtain a series of equations with some unknowns:

- The unknowns are the parameters of our camera matrix

The number of equations increases with the measurements:

- We can minimize the error in these

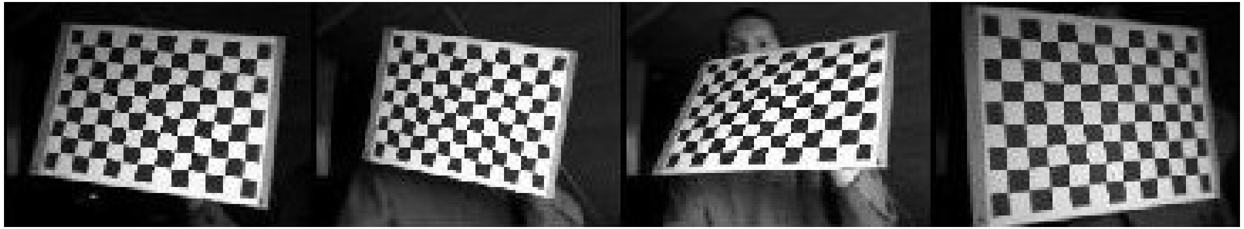
CALIBRATION[5]

Estimating the parameters from a single image is not smart:

- Robustness is low as there is inaccuracy in the localization of the crossings

With multiple images, one obtains a series of equations

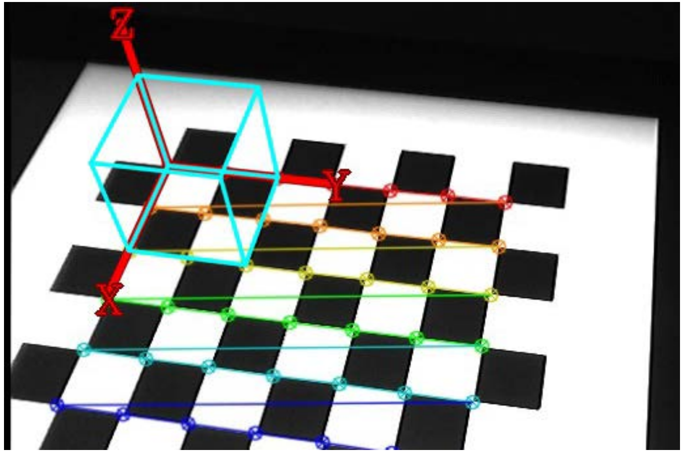
- More images → more robust parameter estimation
- Usually solved by iterative minimization of error function



CALIBRATION[6]

Intrinsic and external parameters are estimated simultaneously

- There is no way to measure 3D camera coordinates!



QUESTIONS?

CAMERA RADIOMETRY

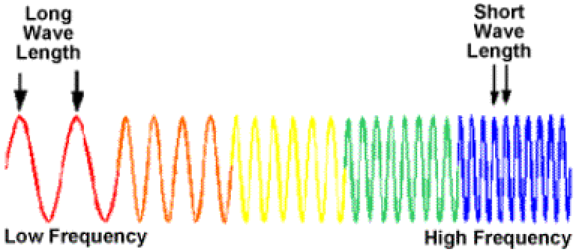
FROM LIGHT TO IMAGE

Digital cameras convert light energy to electric charges

- Light has an intensity (amplitude) and a wavelength
- Intensity determines the strength of the charge
- Wavelength determines the color

Digital cameras use a grid of light-absorbing sensors

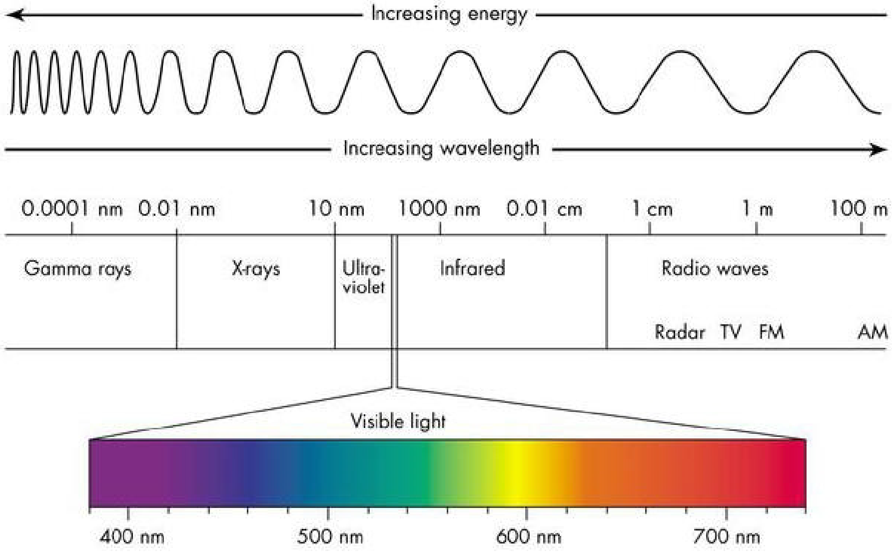
- Electric charges are converted to "intensity" values
- Each sensor is sensitive to range of wavelengths



WAVELENGTH

Humans are sensitive to wavelengths of 380-740nm

- Cameras are typically tuned to this range as well
- Near-infrared (Kinect) or infrared (thermal) sometimes used



WAVELENGTH^2

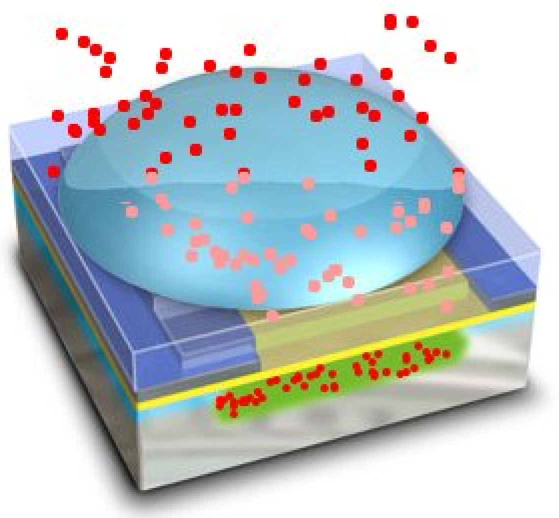
Near-infrared vs. infrared

- Both: invisible for human eye
- Infrared: “heat” signature

RECORDING LIGHT

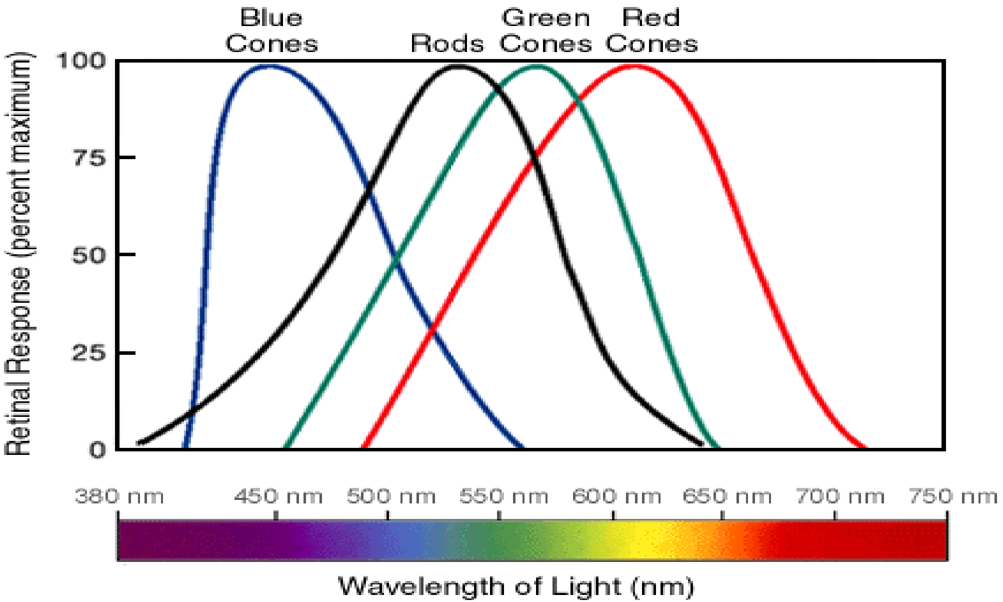
Cameras use CMOS or CCD, grids of imaging sensors

- Convert light energy to electric charges
- Is mono-chromatic, only intensity taken into account



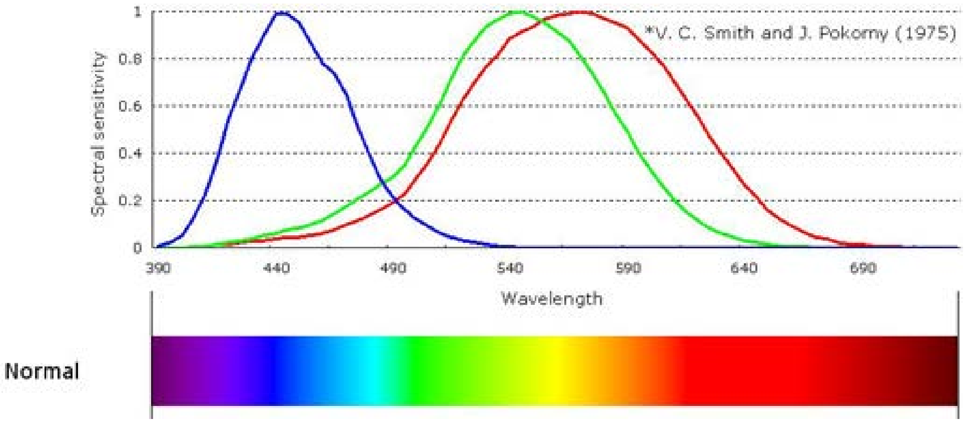
RECORDING COLOR

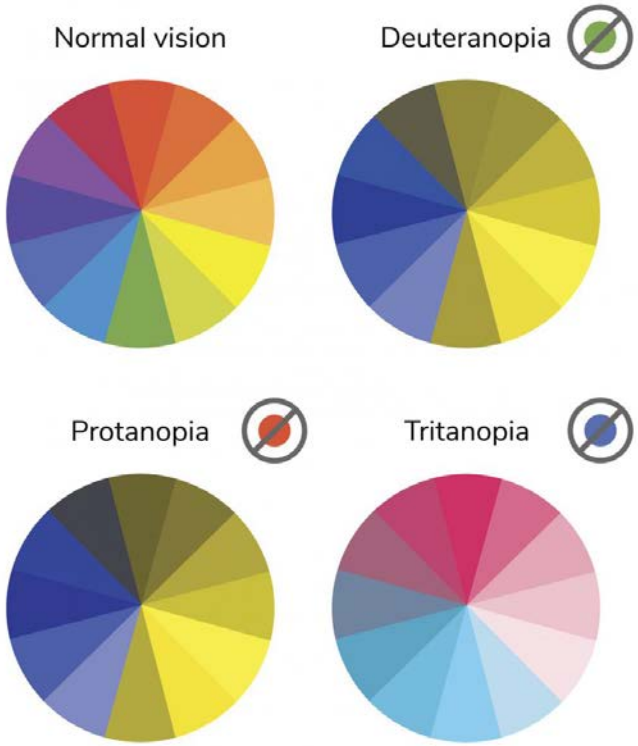
When it comes to perceiving color, CCD works just like cones in the human eye



RECORDING COLOR[2]

Color blindness explained:

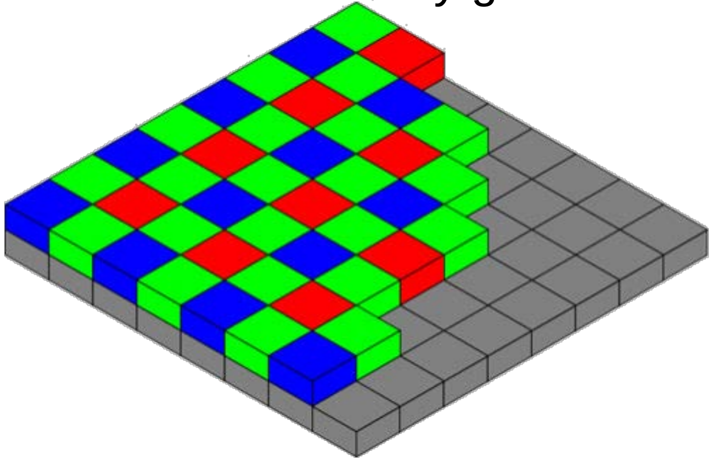


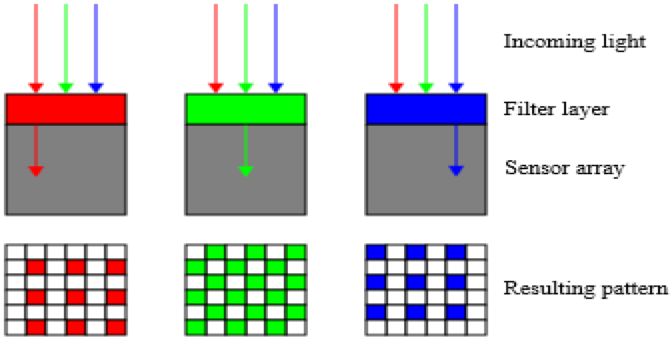


RECORDING COLOR²

To record color, several sensors are used

- Each sensitive to another range of colors
- Color is determined as function of sensor values in a region
- Human eye is more sensitive to luminance, which is mostly determined by green

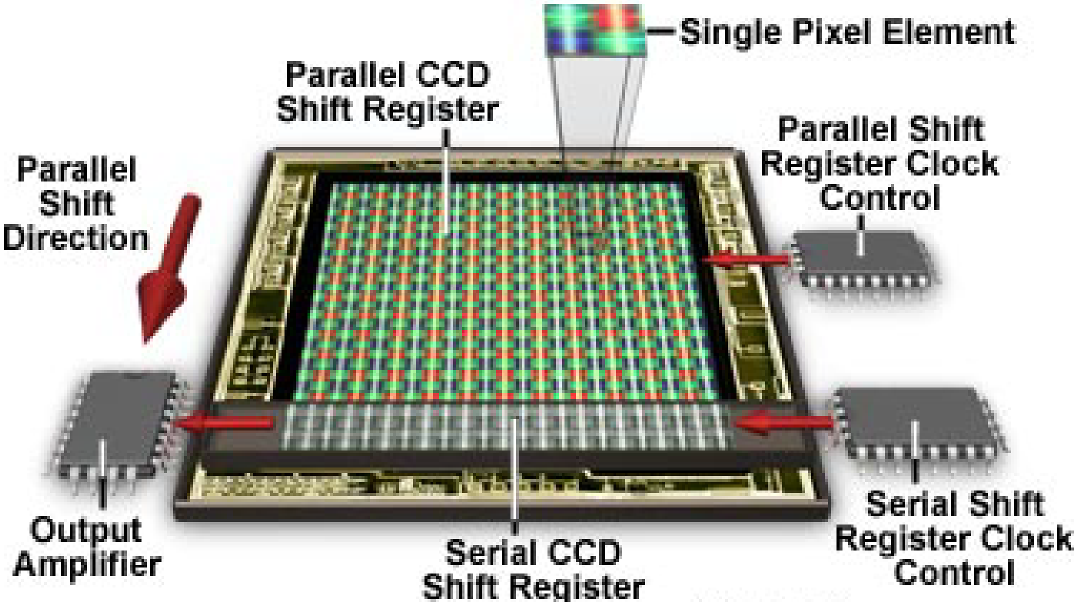




CCD

CCD sensors consist of grid of cells

- Electric charge values read at specific time



CCD^2

Outputs eventually transformed to discrete pixel values

- Different cameras have different color function
- Automatic white-balance control can be achieved
- Post-processing can be used (noise reduction, contrast)

CAMERA ISSUES

EXAMPLE

A projected image can be distorted in many ways:

- Radial distortion
- Vignetting
- Interlacing
- Motion blur
- Chromatic aberration

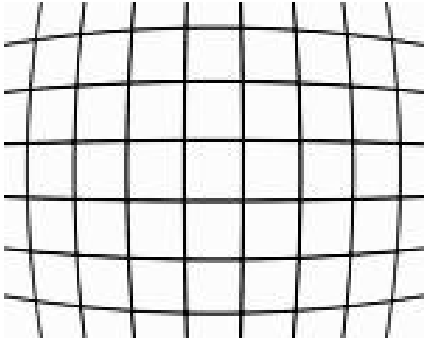
RADIAL DISTORTION

Perspective projection is linear, straight lines in 3D are also straight in the projection.

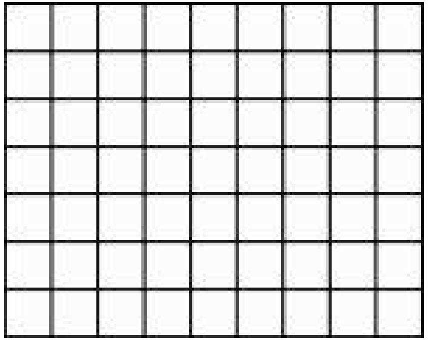
- Not the case when using wide-angle lenses
- These lenses cause radial distortion

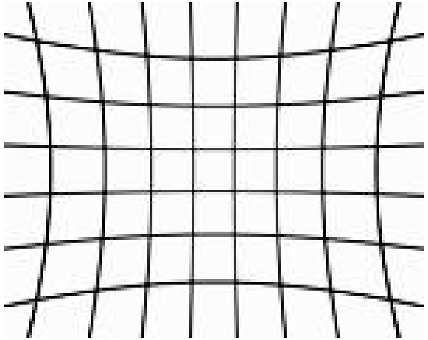
RADIAL DISTORTION[2]

Radial distortion can be corrected, once the camera properties are known



Barrel distortion
(Fish-eye lens)





Pincushion distortion

VIGNETTING

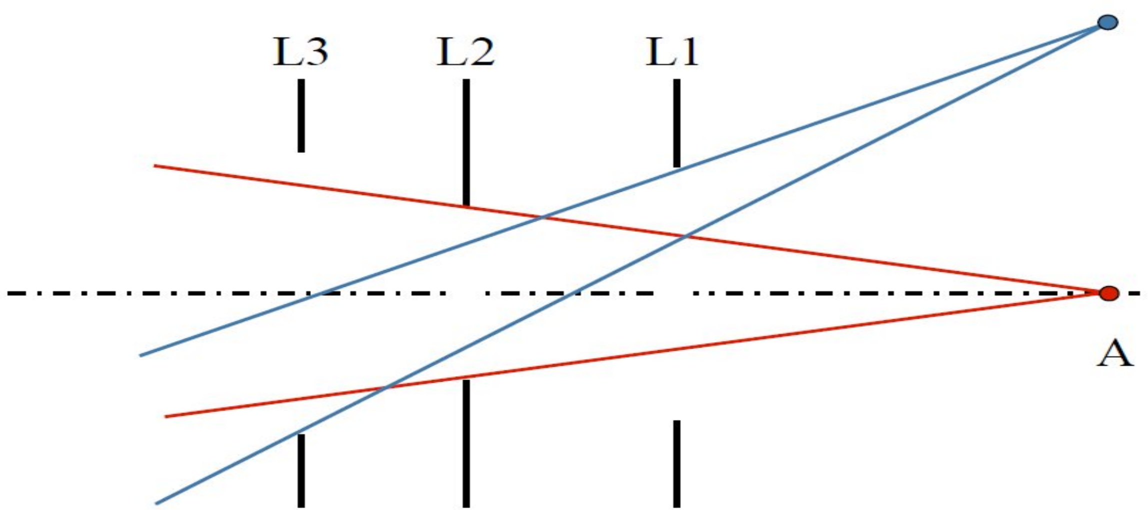
Effect that pixels further from the center are darker

- Typically used as an (Instagram) effect

VIGNETTING[2]

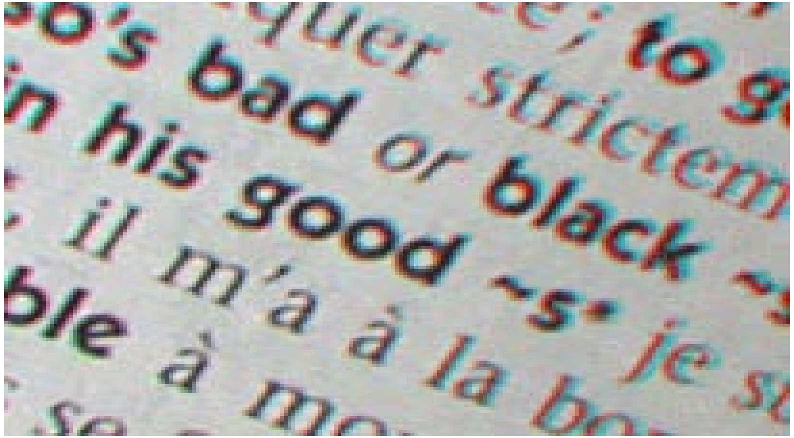
Two causes:

- Natural: these pixels receive less light due to large off-angle
- Mechanical: due to construction of the camera's lens system



CHROMATIC ABERRATIONS

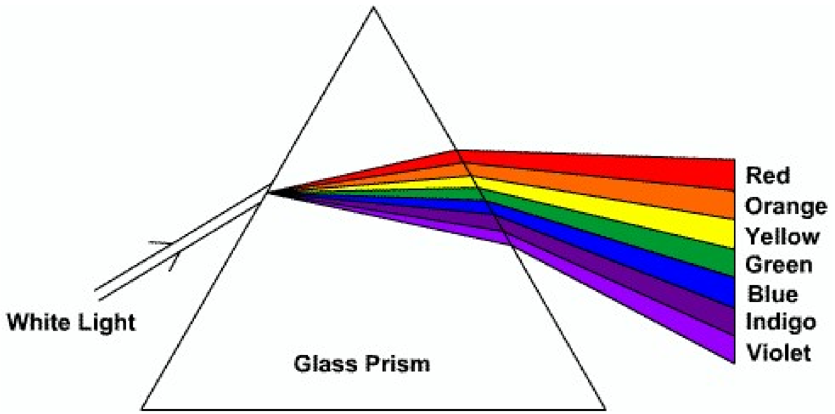
Light rays with different wavelengths have slightly different refraction indices

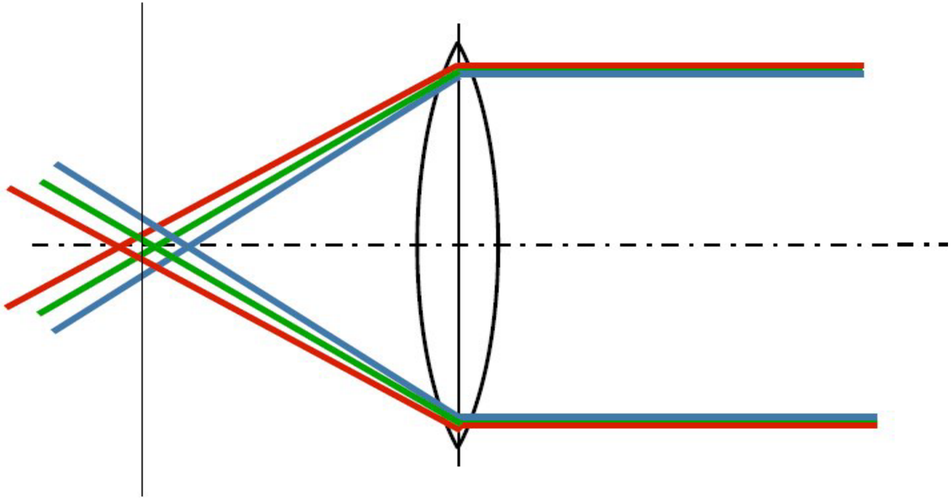


CHROMATIC ABERRATIONS[2]

Difficult to solve as refraction index depends on wavelength, as does the electric charge

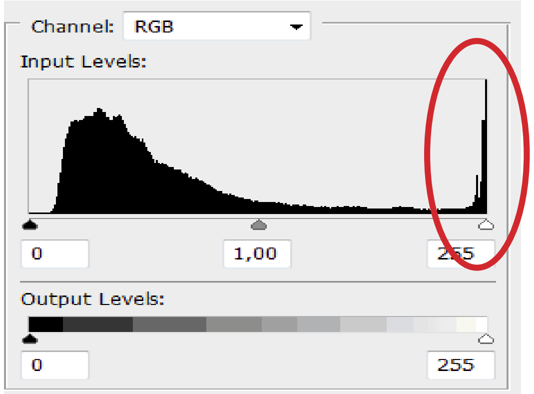
- Typically solved by using compound lenses





OVERSATURATION

Sometimes there is too much light on the sensor, which causes pixels to have the maximum brightness



COLOR SPACES AND DISTANCES

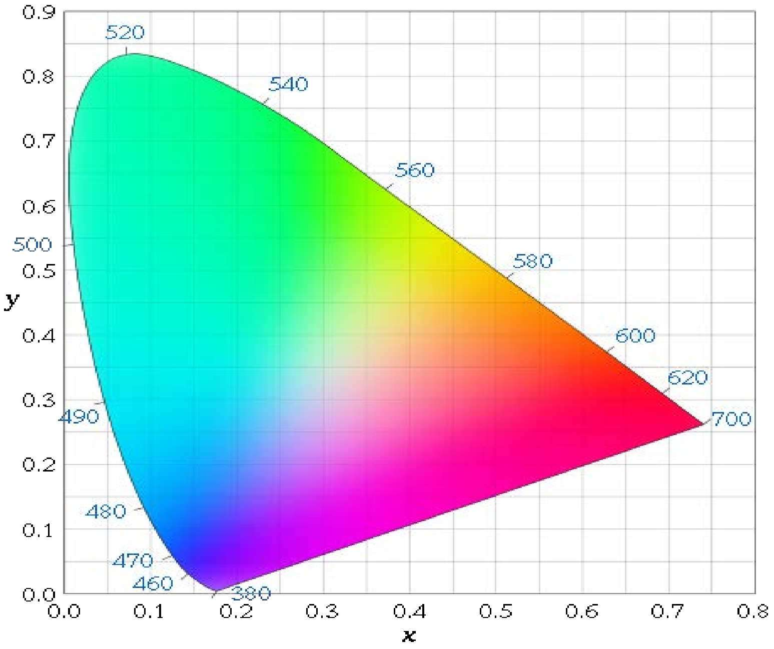
COLORS AND COLOR SPACES

Light has a wavelength and an amplitude

- Wavelength corresponds to color
- Amplitude corresponds to intensity

Wavelength corresponds to color, but:

- Most light is a mixture of wavelengths
- What about intensity?
- What about black?



RGB

RGB (red, green, blue) is an additive color space

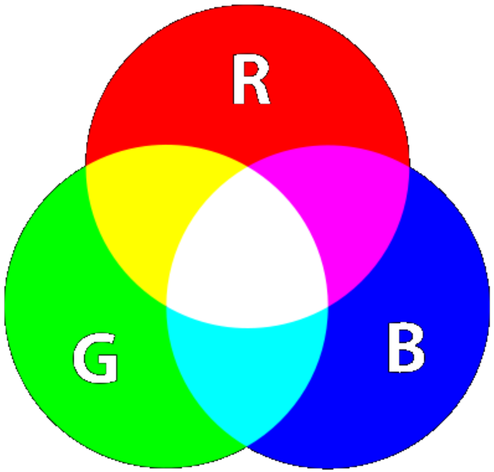
- Three channels → three dimensions

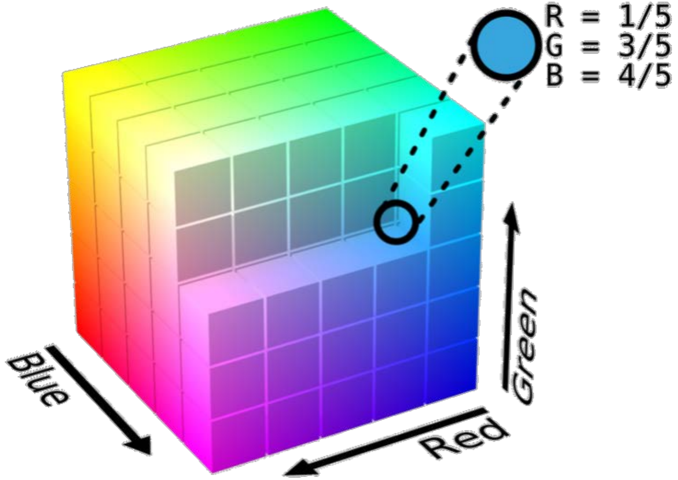
Widely used in:

- Imaging sensors (CMOS/CCD)
- Projectors

Correlation between channels!

- Intensity affects all three channels



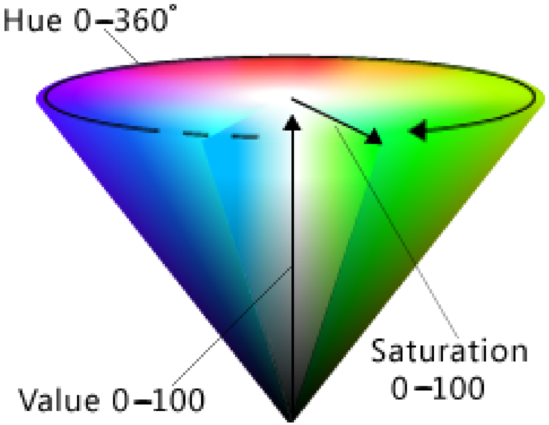


HSV

HSV stands for:

- Hue: color tone
- Saturation: “amount of color”
- Value: intensity

Cone instead of box



No physical or perceptual background but:

- Color corrections can be made easily
- Intensity has separate channel
- Saturation provides less information and can often be ignored

CIELAB

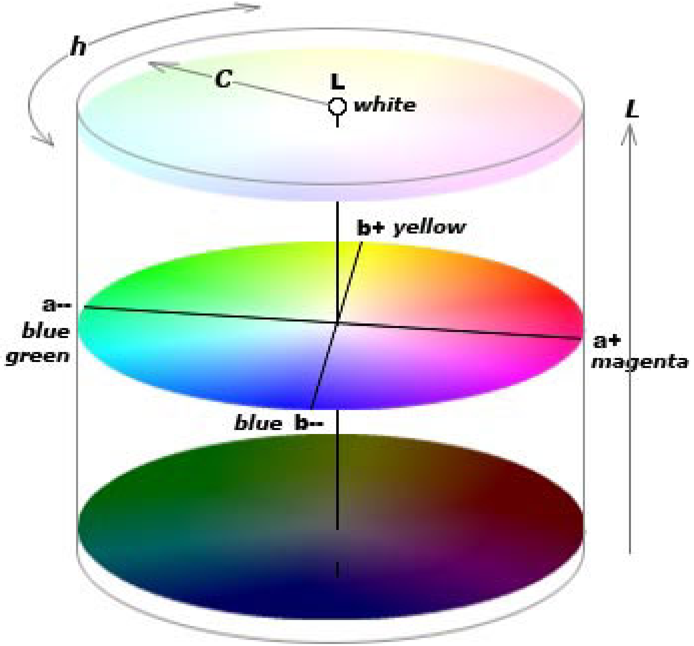
CIELab has three channels:

- L: intensity
- A*: green – red
- B*: blue – yellow

Cylindrical model:

- Color in 2D plane

Good for color corrections



DISTANCES

The distance in color value between pixel *A* and *B* can be expressed in different ways

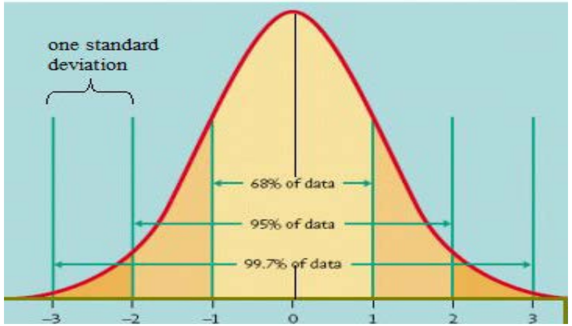
- Single channel: $d = |A - B|$

- Three channels, summed (Manhattan): $d = \sum_{c=1}^{3} |\boldsymbol{A_c} - \boldsymbol{B_c}|$

- Three channels, shortest distance (Euclidian/Pythagoras):

$$d = \sqrt{\sum_{c=1}^{3} (\boldsymbol{A_c} - \boldsymbol{B_c})^2}$$

DISTANCES^2



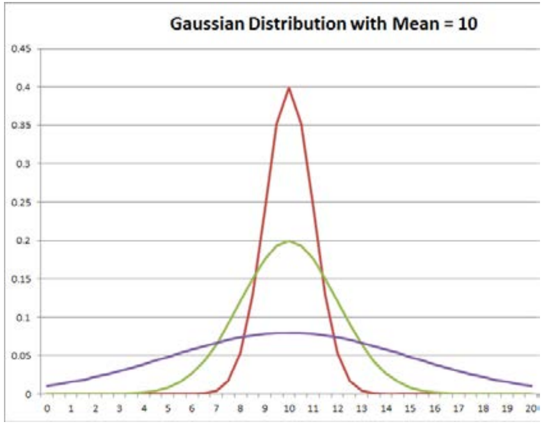
These measures are between two pixels/colors

If we compare a pixel to a group of pixels, we can use color variance:

- Typically, a normal/Gaussian distribution is assumed
- Mean and variance modeled

When setting a threshold on the distance:

- Low variance → stricter threshold
- More variance → more forgiving threshold



DISTANCES[3]

Mahalanobis distance (channel mean $\boldsymbol{\mu}_c$ and standard deviation $\boldsymbol{\sigma}_c$):

$$d = \sqrt{\sum_{c=1}^{3} \frac{(\boldsymbol{A_c} - \boldsymbol{\mu_c})^2}{\boldsymbol{\sigma_c}^2}}$$

Color channels can also be treated differently:

- Omit channels from the distance
- Have different thresholds (scale distances differently per channel)
- Have a threshold that is ratio of value per channel (e.g. intensity in HSV)

TAKE-HOME MESSAGE

There are different color spaces

- A color can be described in each of them
- Each color space has different (dis)advantages
- "Best" color space depends on application, scene, etc.

Distances between colors can be measured in different ways

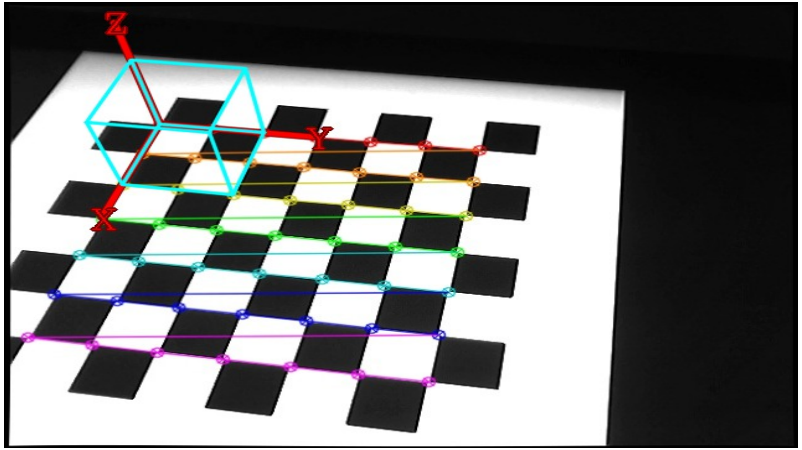
- Between means: single-channel, Manhattan, Euclidian
- Between distribution and point: Mahalanobis

ASSIGNMENT

ASSIGNMENT

Geometric calibration using OpenCV:

- Gentle introduction to OpenCV and C++
- Needed for Assignment 2
- Uses standard OpenCV functions (browse these!)



ASSIGNMENT^2

There is an “offline” and an “online” part in the assignment.

Offline:

- Determine camera parameters using the photos of your chessboard
- Save those parameters

Online:

- Load the camera parameters (DON’T calculate them again)
- Load an image (or read from your camera) and draw the cube

ASSIGNMENT[3]

Workflow offline:

- Print checkerboard on piece of paper (or take a real one)
- Measure the stride length and fill it in
- Take a number of pictures with the camera
- Determine camera parameters using OpenCV functions (browse!)
- Now your camera is calibrated

Workflow online:

- Read an image/camera frame
- Draw a box on a detected chessboard in the right perspective

ASSIGNMENT[4]

Software:

- OpenCV: http://opencv.org/
- Linux, Windows or Mac

Deadline Sunday February 17, 23:00. Strict!

- In pairs. Can't find a partner: come to me after the lecture
- Submit code online

Breixo will supervise the assignments

- b.solinofernandez@students.uu.nl
- Join Slack: https://join.slack.com/t/infomcv2019/signup

NEXT LECTURE

Volume-based 3D reconstruction

- Using multiple calibrated views to determine shape
- Voxel-based representations
- Basis for Assignment 2, requires Assignment 1
- Chapter 11.6 of Szeliski book

Next Thursday 11:00-12:45, RUPPERT-042

QUESTIONS?