Computer Vision

CS-E4850, 5 study credits

Juho Kannala
Aalto University

Plan for today

- Background
  - What is computer vision?
  - Why to study computer vision?
- Overview of the course
- Lecture 1: Image formation

Credits: Material for slides borrowed from Victor Prisacariu, Andrew Zisserman, Esa Rahtu, James Hays, Derek Hoiem, Svetlana Lazebnik, Steve Seitz, David Forsyth, and others

Course personnel

- Lecturer:
  Juho Kannala
  juho.kannala@aalto.fi

- Main teaching assistant:
  Antti Parviainen
  antti.parviainen@aalto.fi

A few words about me

Juho Kannala

Assistant Professor of Computer vision

- PhD, University of Oulu 2010
- Professor at Aalto since 2016

- Working with computer vision since 2000
- Recent projects and other info available on my homepage: https://users.aalto.fi/~kannalj1/

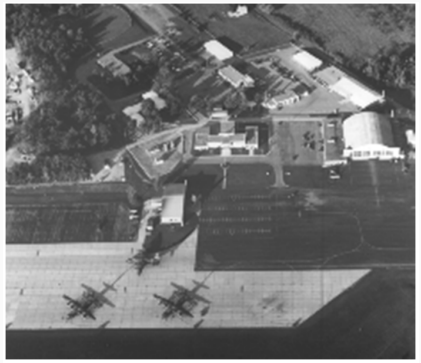
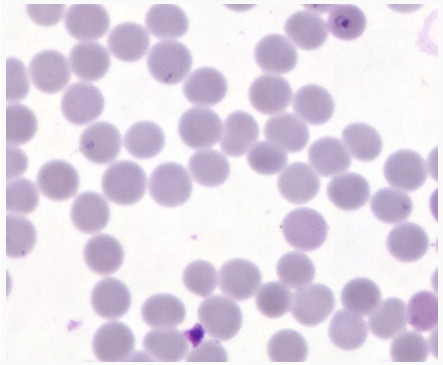
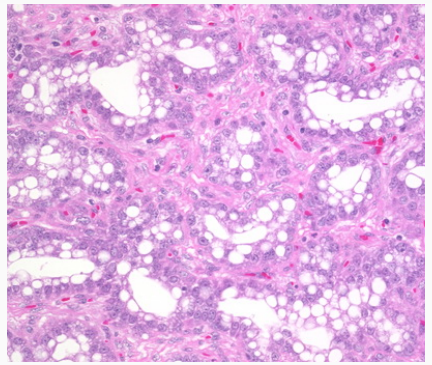
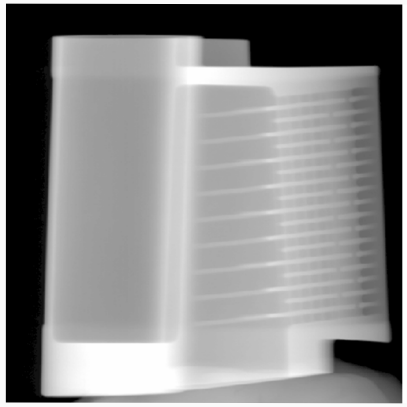
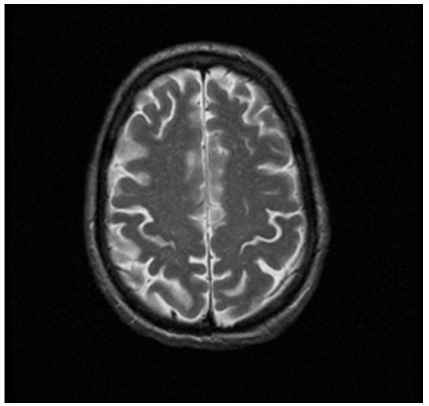
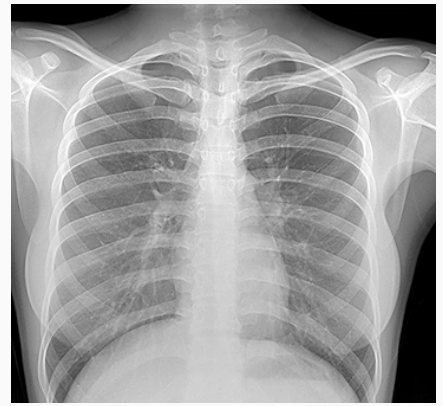
Motivation - what is computer vision?

Make computers understand images

- What kind of scene?
- Where are the cars?
- How far are the buildings?
- Where are the cars going?
- …..

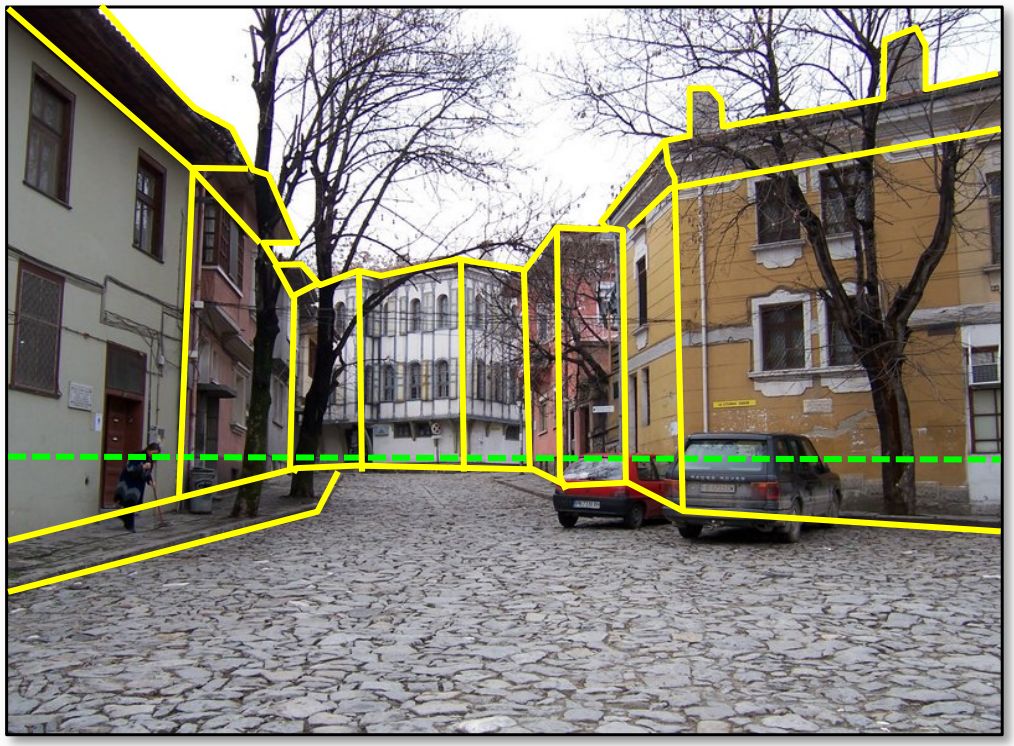
Many data modalities

- 2D or 3D still images
- Video frames
- X-ray
- Ultra-sound
- Microscope
- ….

What kind of information can be extracted?

Semantic information



Geometric information

What do we have here?

Wrong! Very hard big data problem…

- Hardware perspective:
  - RGB stereo images with 30 frames per second -> 100s MB/s data stream.
  - Non-trivial processing per each byte.
  - Massive image collections.
- Mathematical perspective
  - Information is highly implicit or lost by reprojection
  - 2D -> 3D mapping is ill-posed and ill-conditioned -> need to use constraints

Wrong! Very hard big data problem…

- Artificial intelligence perspective
  - Images have uneven information content
  - Computational visual semantics is hard (what does visual stuff mean exactly?)
  - If we have limited time, what is the important visual stuff right now?

Still a massive challenge - if we want genuine autonomy.

Natural vision

- Humans see effortlessly

Natural vision

- Humans see effortlessly, but… it is very hard work for our brains!
  - There are billions of neurons in human brain
  - Years of evolution generated hardwired priors.

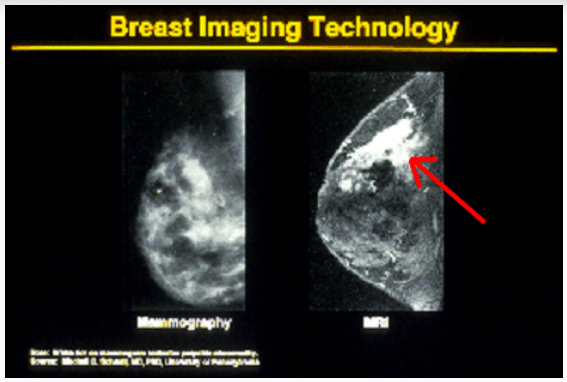
So why bother?
What are the advantages?

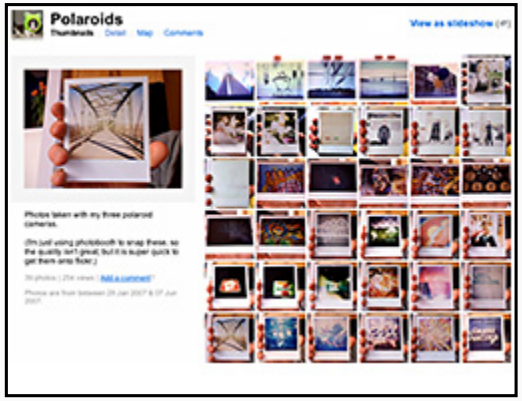
Why computer vision matters?

- Engineering point of view - Computer Vision helps to solve many practical problems: business potential
- Scientific point of view - Human kind of visual system is one of the grand challenges of Artificial Intelligence (AI)
  - AI itself is a grand challenge of computing

Why computer vision matters?

- Safety
- Health
- Security
- Fun
- Access
- ….





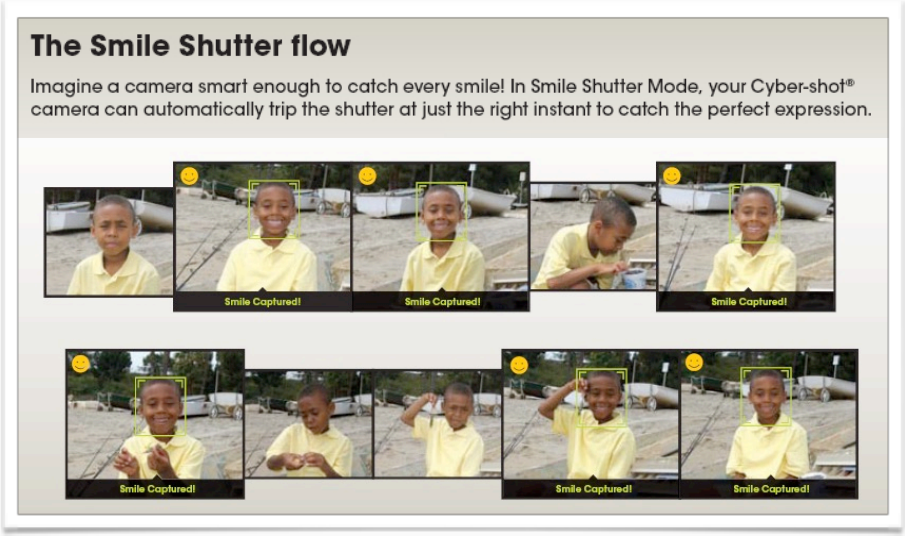
Computer vision is already here

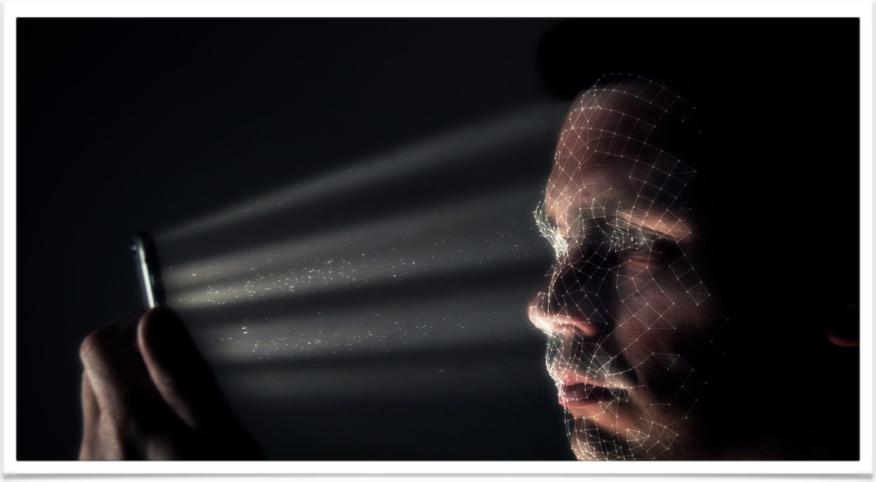
- You are surrounded by devices using computer vision
- Imagine what can be done with already installed cameras!

Motivation - Success stories

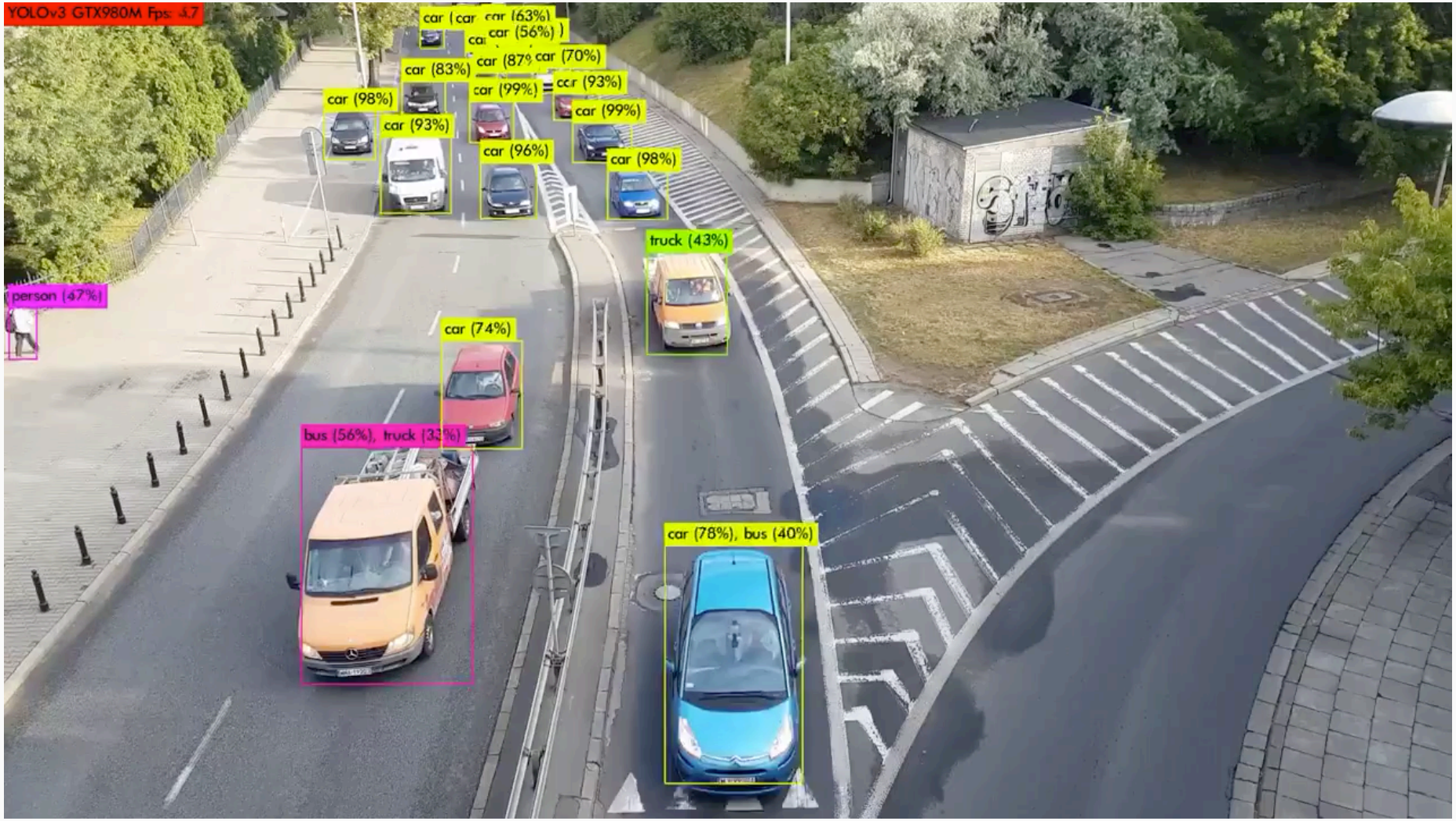
Recognizing “simple” patterns

Face recognition



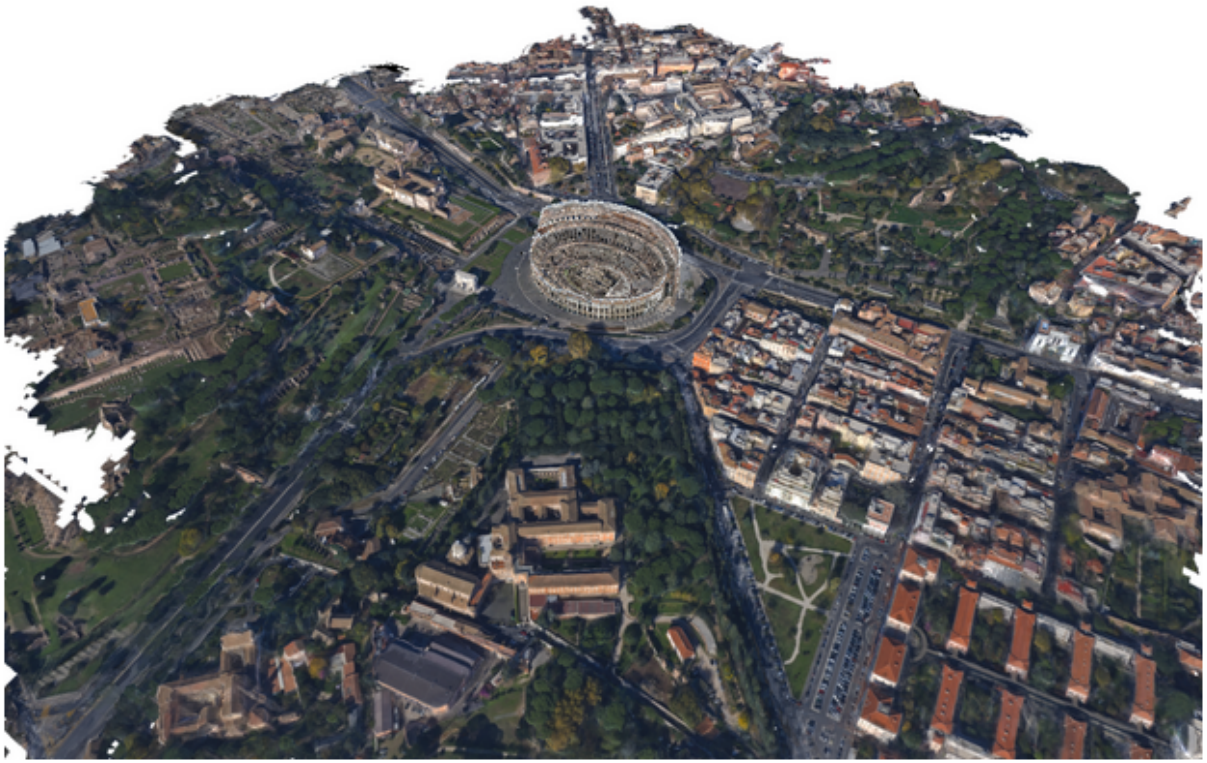


Object detection and recognition



Reconstruction: 3D from photo collections

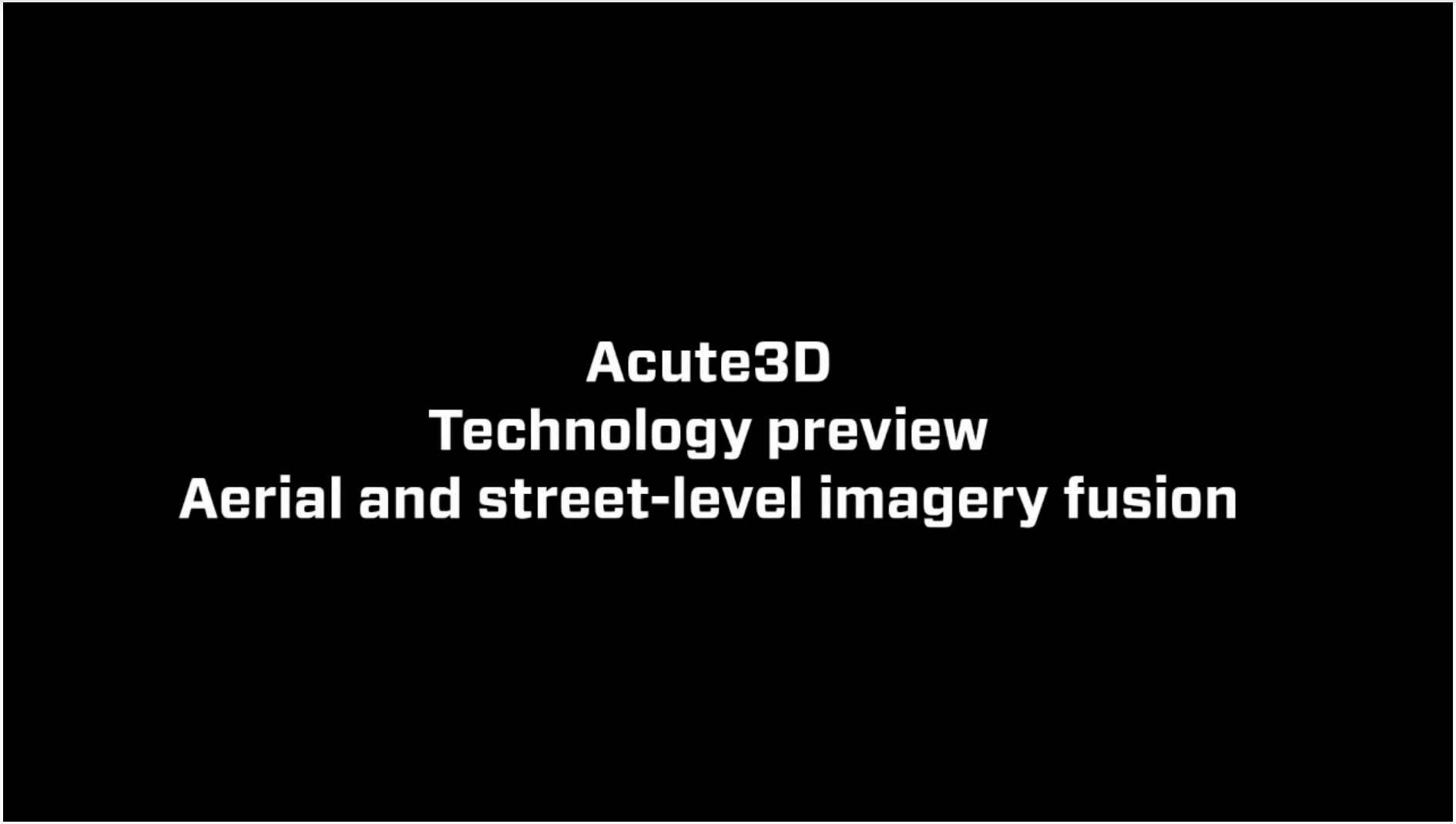
Colosseum, Rome, Italy



San Marco Square, Venice, Italy

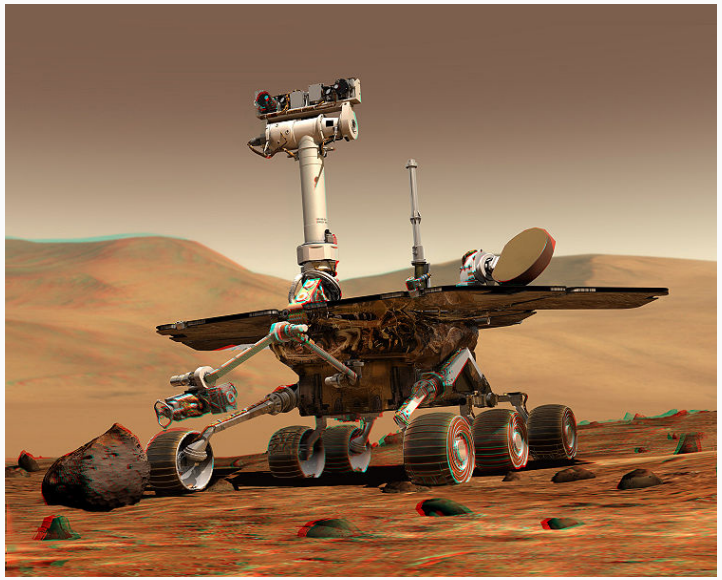
The Visual Turing test for Scene Reconstruction,
Shan, Adams, Curless, Furukawa, Seitz, in 3DV 2013. YouTube video.

A recent commercial 3D reconstruction system

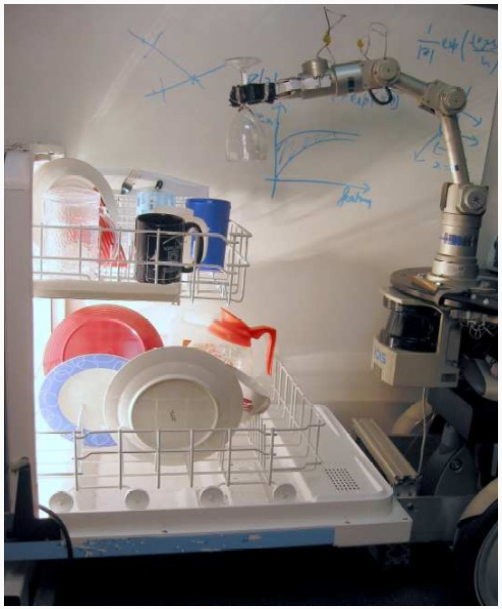


YouTube

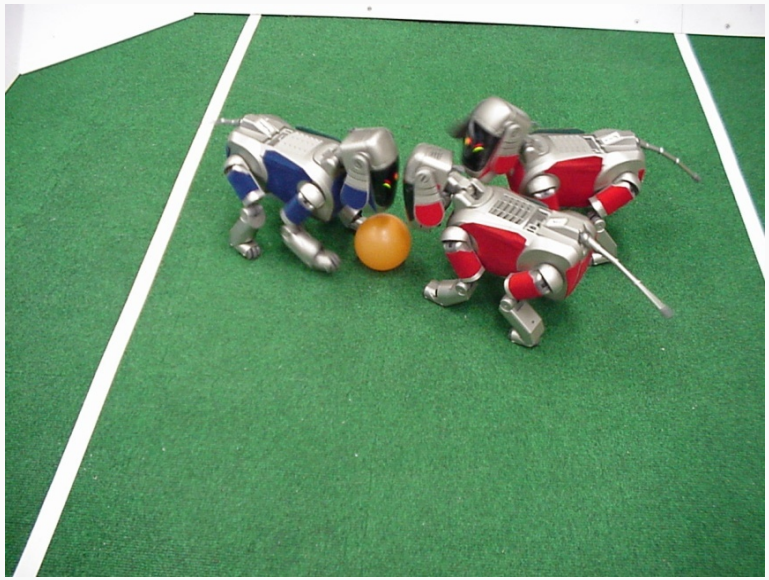
Robotics



NASA's Mars Rover
See "Computer Vision on Mars"

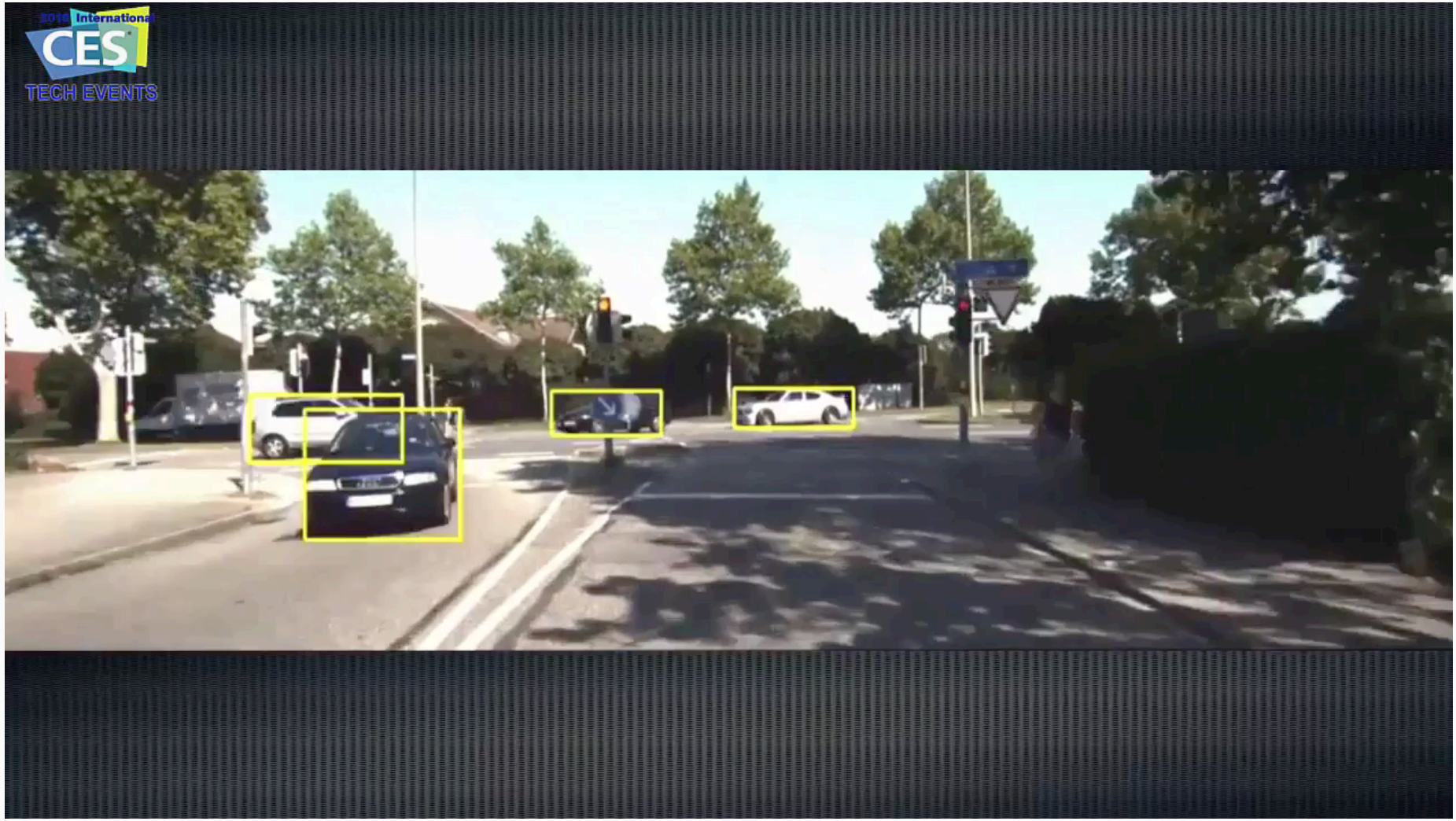


STAIRS at Stanford
Saxena et al. 2008

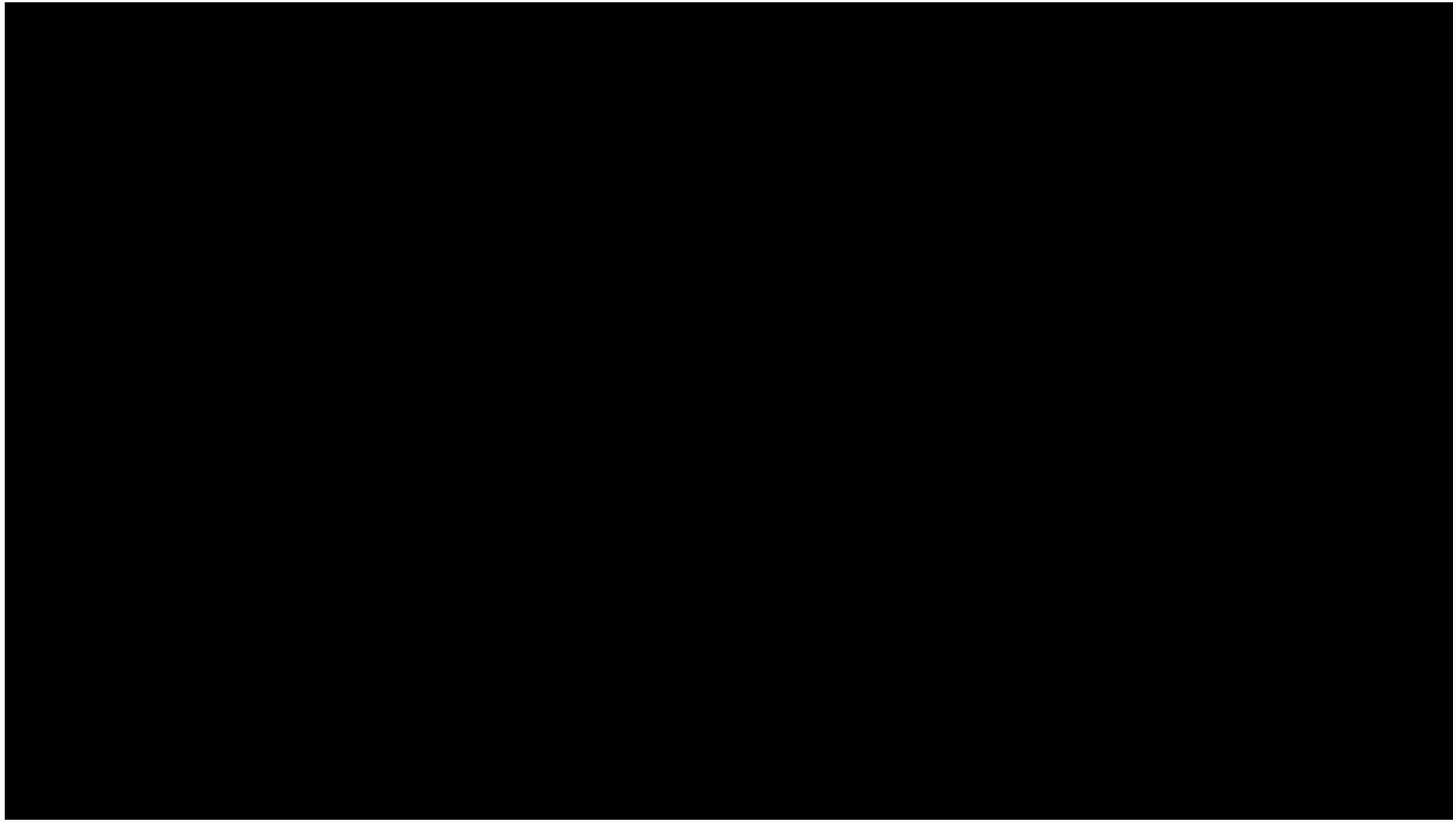


Robocup
See www.robocup.org

Self-driving cars (Nvidia @ CES 2016)



Visual odometry and SLAM



Augmented Reality (AR) and Virtual Reality (VR)

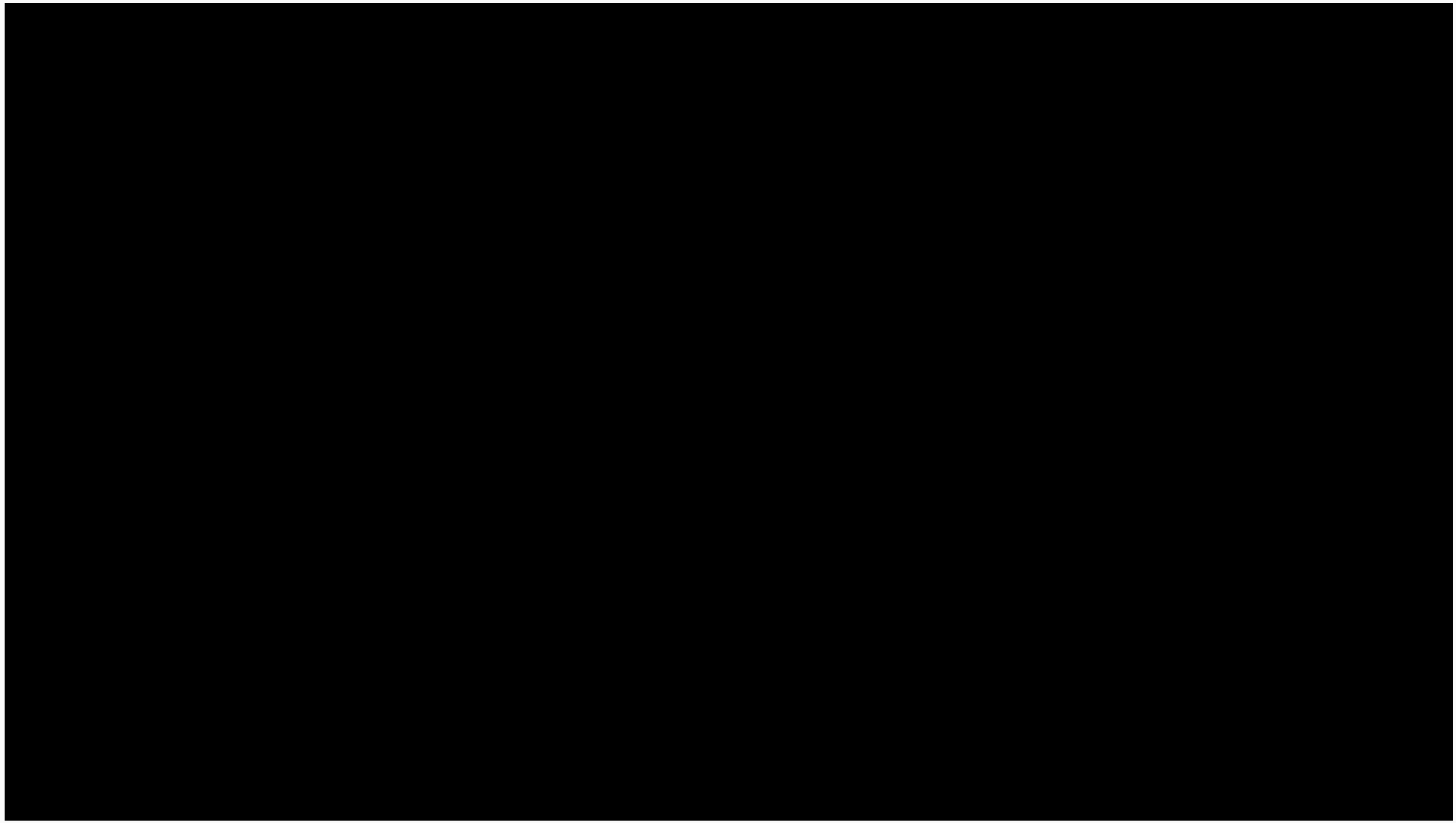
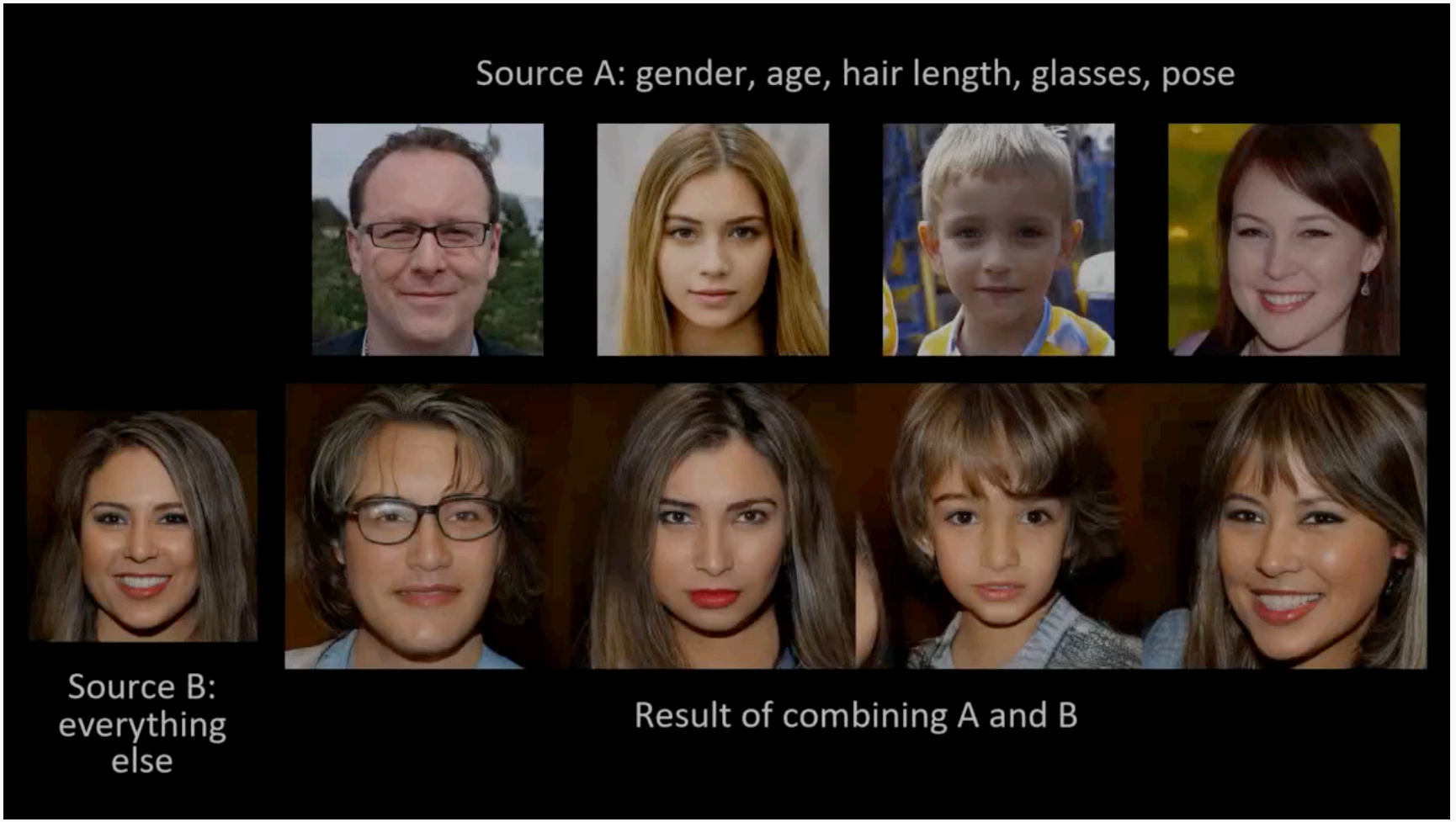


Image generation

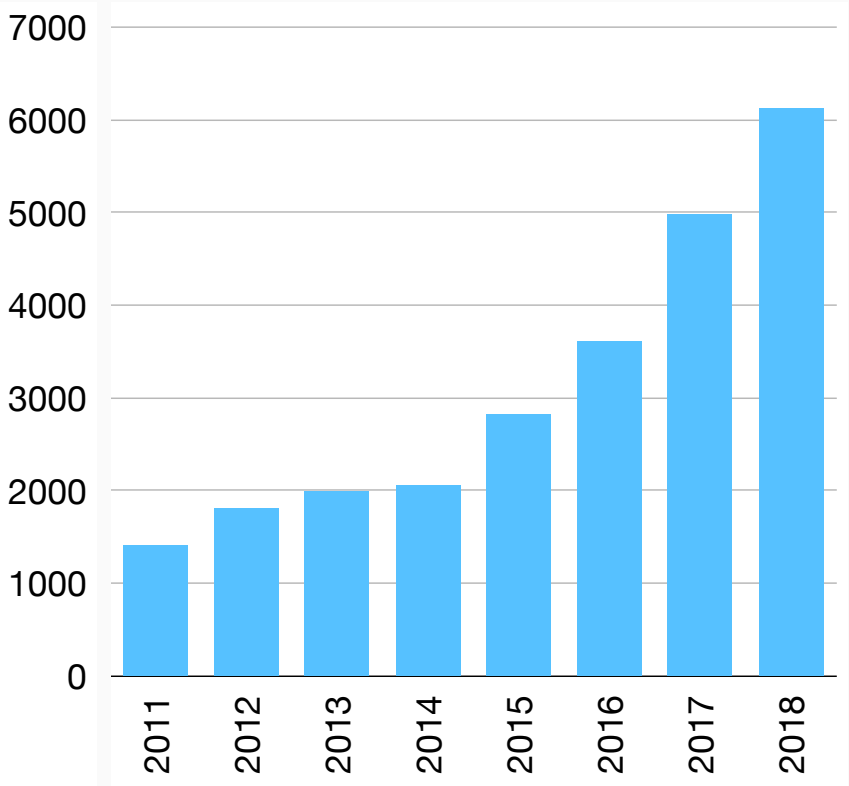


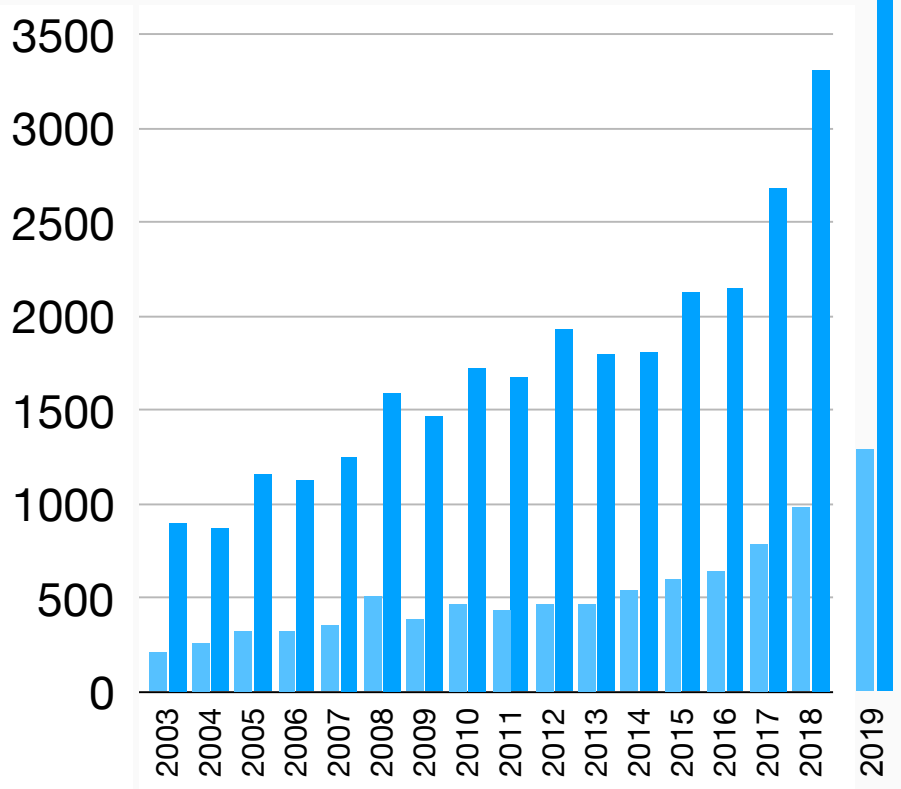
A style-based generator architecture for generative adversarial networks. Karras, Laine, Aila. CVPR 2019.

Current state of the affairs

- Many of the previous examples are less than 5 years old!
- Many new applications to appear in the next 5 years
- Strong open source culture
  - Many recent state-of-the-art methods are freely available
  - See papers from top conferences like CVPR, ECCV, ICCV, and NeurIPS

Rapidly growing area





Attendees and submissions to IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Rapidly growing area

	Publication	h5-index	h5-median
1.	Nature	362	542
2.	The New England Journal of Medicine	358	602
3.	Science	345	497
4.	The Lancet	278	417
5.	Chemical Society reviews	256	366
6.	Cell	244	366
7.	Nature Communications	240	318
8.	Chemical Reviews	239	373
9.	Journal of the American Chemical Society	236	309
10.	Advanced Materials	235	336
11.	Proceedings of the National Academy of Sciences	226	291
12.	Angewandte Chemie International Edition	213	295
13.	JAMA	209	309
14.	Nucleic Acids Research	208	392
15.	ACS Nano	199	279
16.	Physical Review Letters	197	286
17.	Energy and Environmental Science	196	330
18.	Journal of Clinical Oncology	196	279
19.	Nano Letters	194	281
20.	IEEE Conference on Computer Vision and Pattern Recognition, CVPR	188	302

Ref. Google Scholar top publications.

Rapidly growing area - substantial commercial interest

CVPR 2018 sponsors

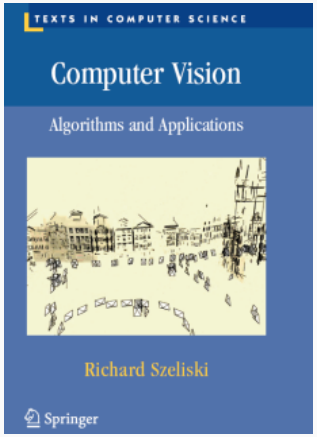
Plenty of job opportunities

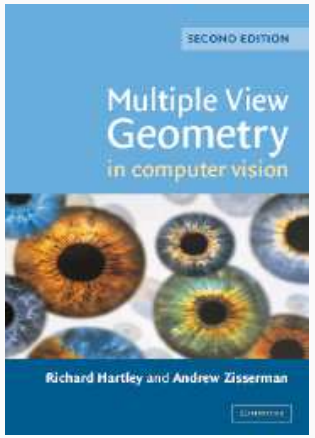
- Companies are looking for computer vision and deep learning experts.
- Big Internet players are investing heavily (Apple, Google, Facebook, Microsoft, Baidu, Tencent, …) as well as car industry (Tesla, BMW,…)
- Strong imaging ecosystem also in Finland

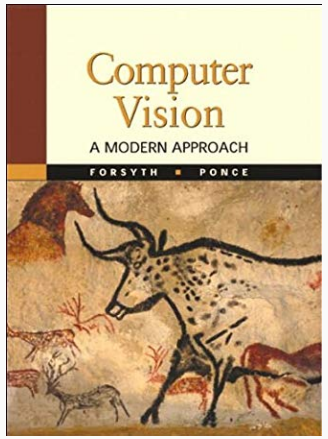
Specifics of this course

Course textbooks

- Szeliski: Computer Vision
  - Full-copy freely available
- Hartley & Zisserman: Multiple View Geometry in Computer Vision
  - Available as an e-book via library
- Forsyth & Ponce: Computer Vision
  - Full-copy freely available







What will you learn on this course?

- Tentative course content (numbers refer to chapters in Szeliski's book):
  - Image formation and processing (2, 3)
  - Feature detection and matching (4)
  - Feature based alignment and image stitching (6,9)
  - Optical flow and tracking (8)
  - Structure from motion, stereo and 3D reconstruction (7, 11, 12)
  - Basics of image classification and convolutional neural networks
  - Object recognition and detection (14)

What will you NOT learn on this course?

- Software packages
  - PyTorch, TensorFlow, Keras, Caffe, etc.
  - We have simple exercises with Python/Matlab though
- In-depth deep learning
  - Tweaking architectures, loss functions, etc.
  - Note that there exists a separate deep learning course (CS-E4890)
- All the bells and whistles in the state-of-the-art systems
  - We concentrate on the basic concepts (get them right and the rest is easier for you)

Requirements

- Get more than 0 points from at least 8 exercise rounds
  (i.e. solve at least 1 task from 8 different weekly rounds)
- Pass the exam

Hints

- Doing homework takes time but is often a good way to learn in depth
- Try to do more than the minimum - homework points are taken into account in the grading
- Still, solving all homework tasks is not required for a good grade if you do well in the exam
- Note that the amount of work varies a bit between weeks - exercises are published early so that you can do them in advance if needed

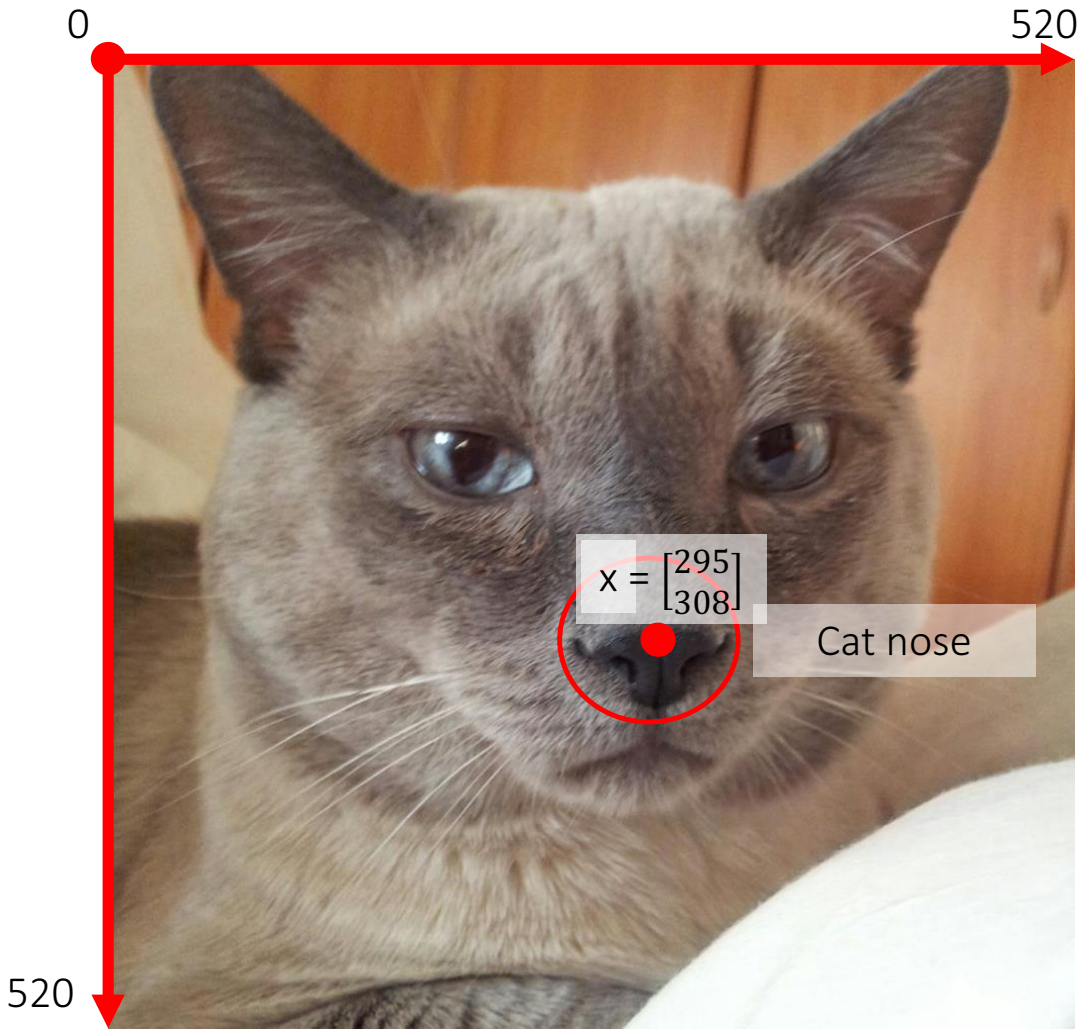
Questions at this point?

Lecture 1: Camera model

Relevant reading

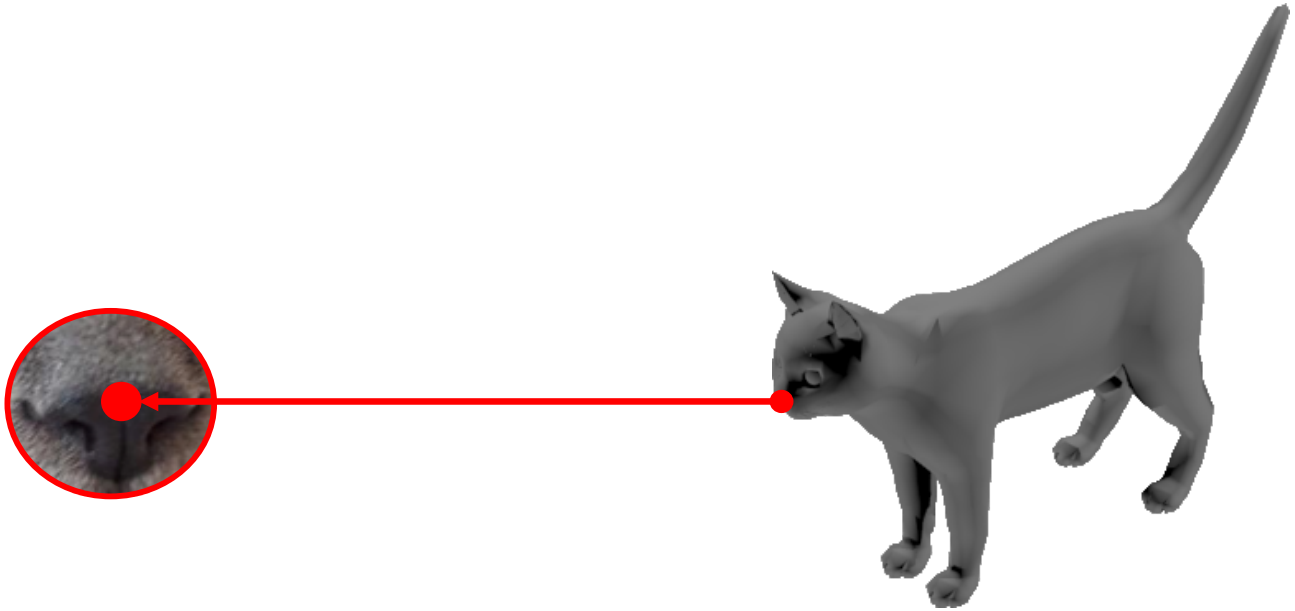
- Chapters 2, 3, and 6 in [Hartley & Zisserman]
  - Comprehensive presentation of the core content
- Chapter 2 in [Szeliski]
  - Broader overview of the image formation

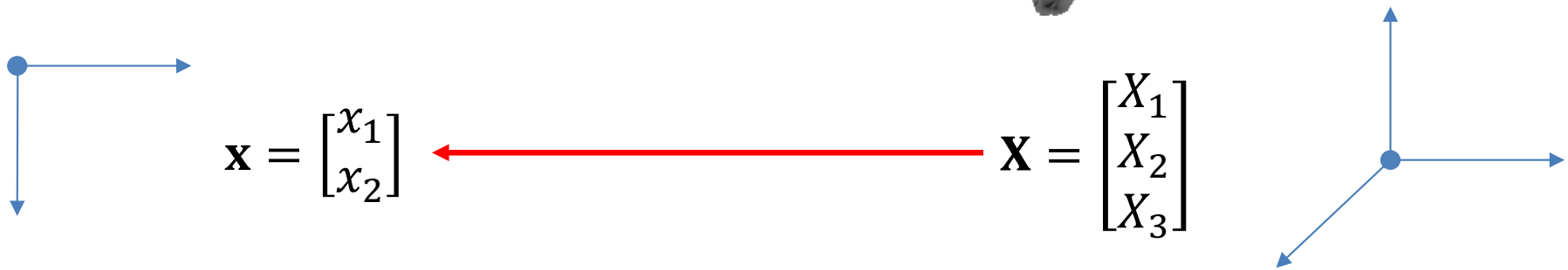
This is (a picture of) a cat



Credits: Victor Prisacariau

Cat lives in a 3D world

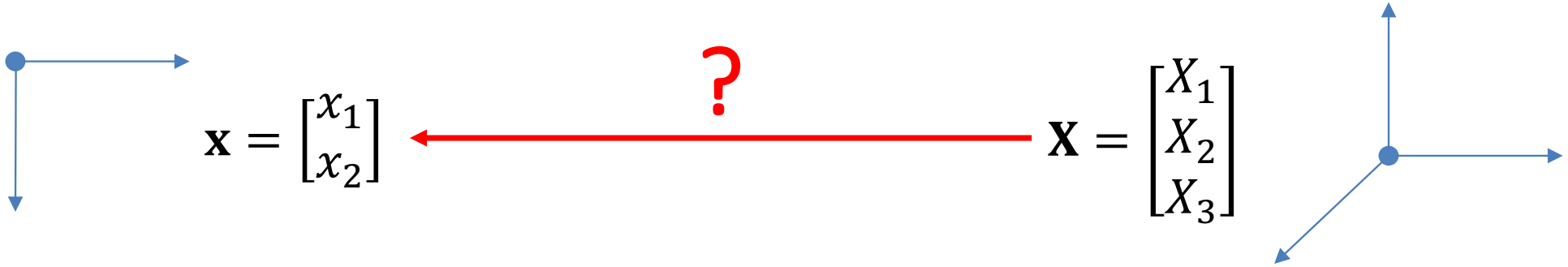


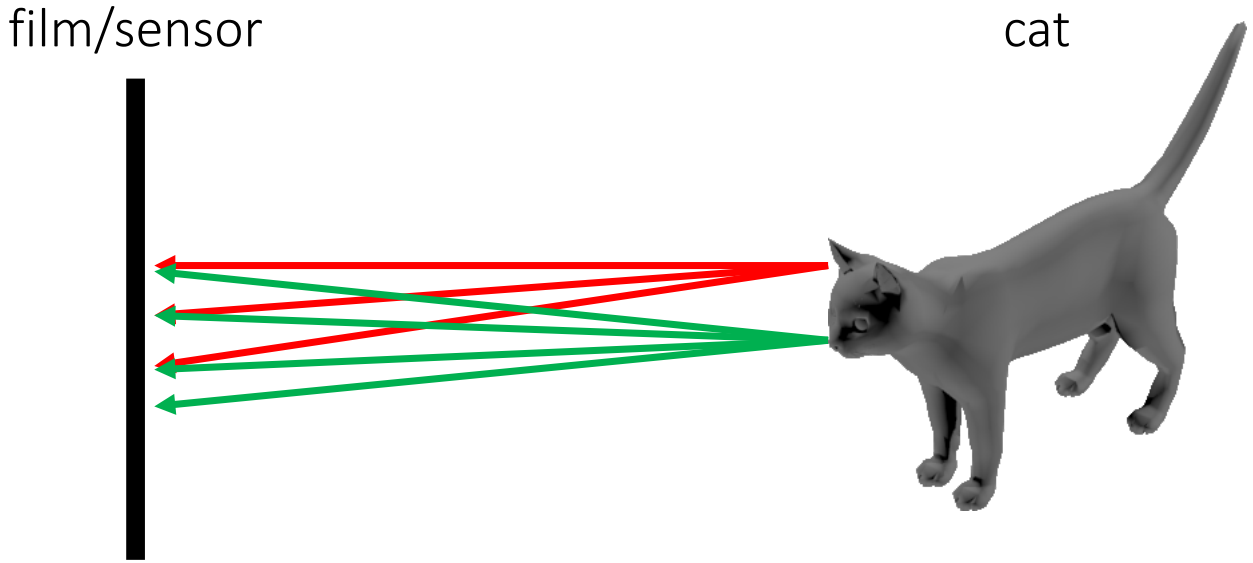


The point **X** in world space projects to the point **x** in image space.

Credits: Victor Prisacariau

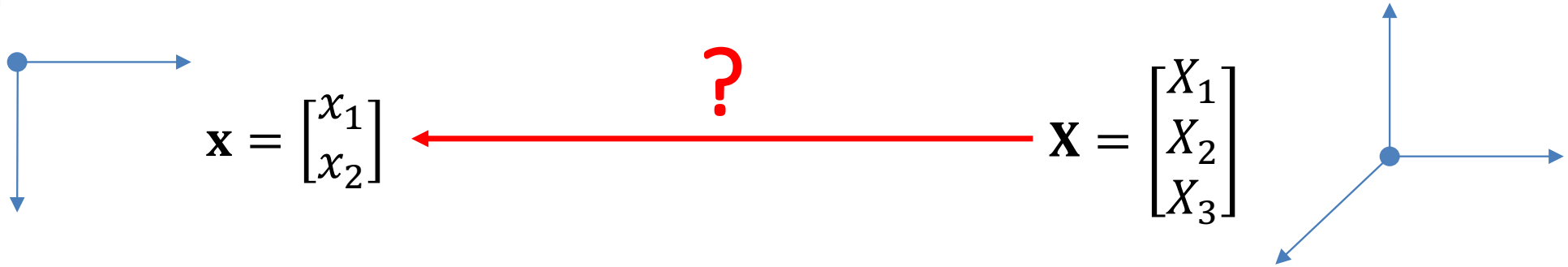
Going from X in 3D to x in 2D

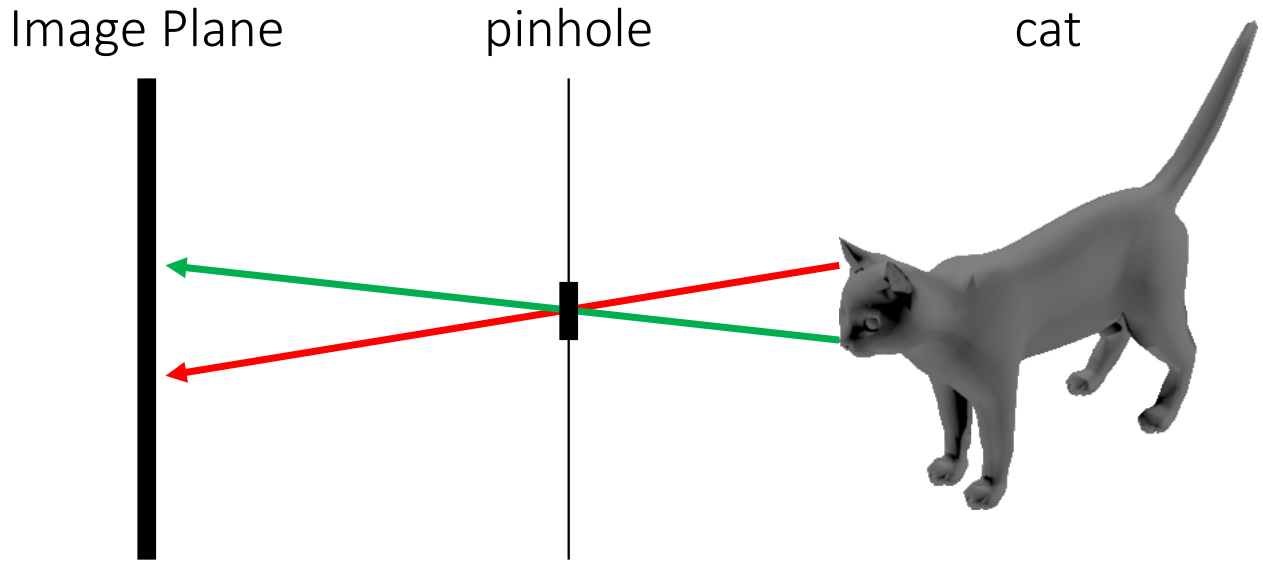




The output would be blurry if film just exposed to the cat.

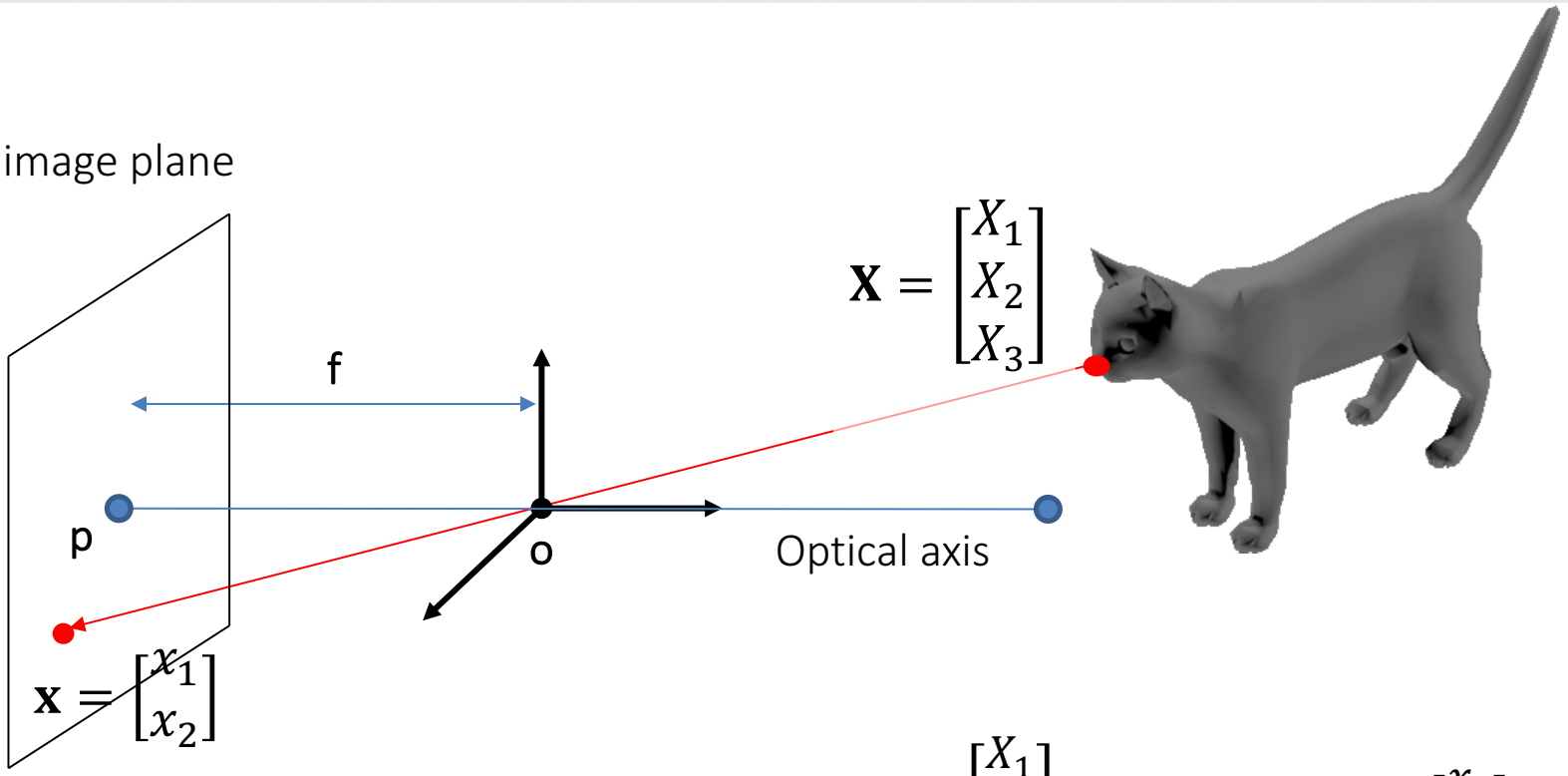
Pinhole camera





All rays passing through a single point (center of projection)

Pinhole camera

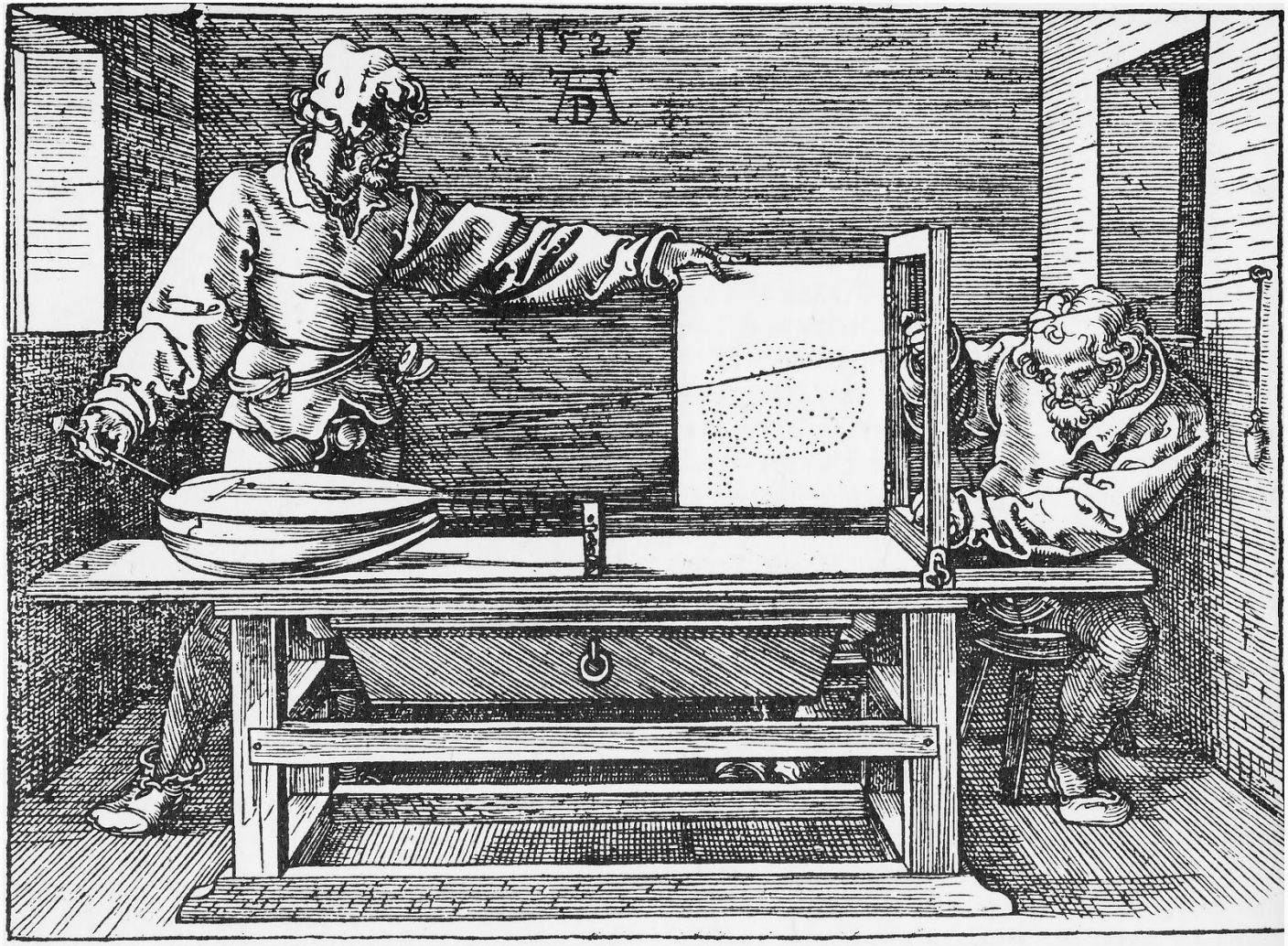


f – focal length
o – camera origin
p – principal point

The 3D point $\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}$ is imaged into $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ as:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} f\frac{X_1}{X_3} \\ f\frac{X_2}{X_3} \end{bmatrix}$$

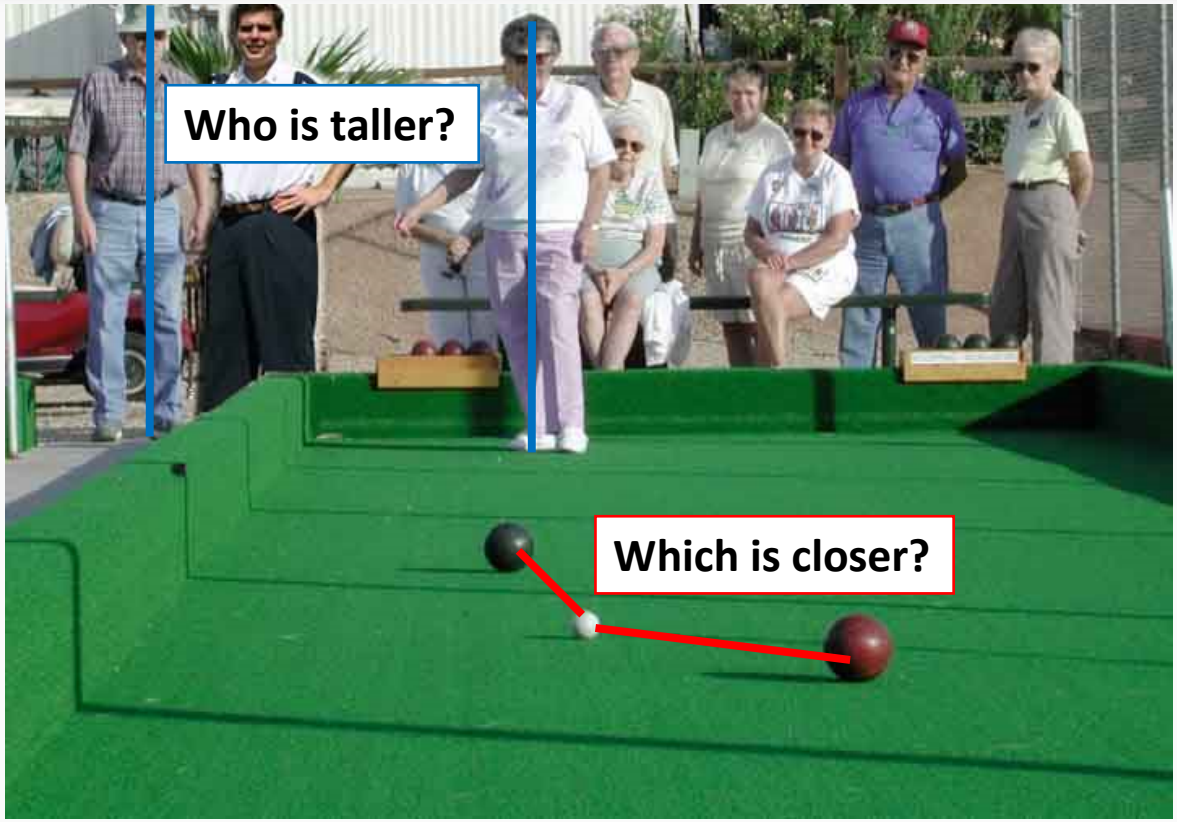
Pinhole camera



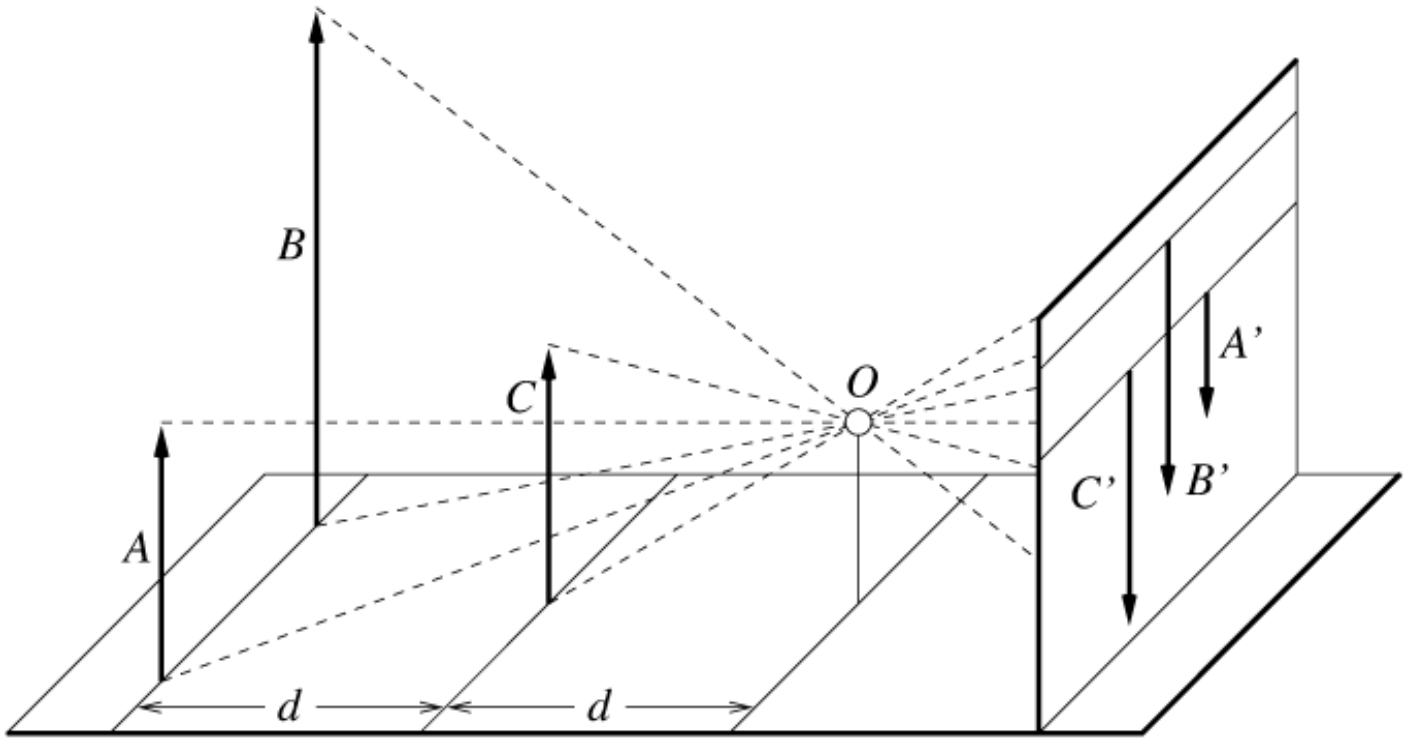
What happens in the projection?

- Projection from 3D to 2D -> information is lost
- What properties are preserved?
  - Straight lines
  - Incidence
- What properties are not preserved?
  - Angles
  - Lengths

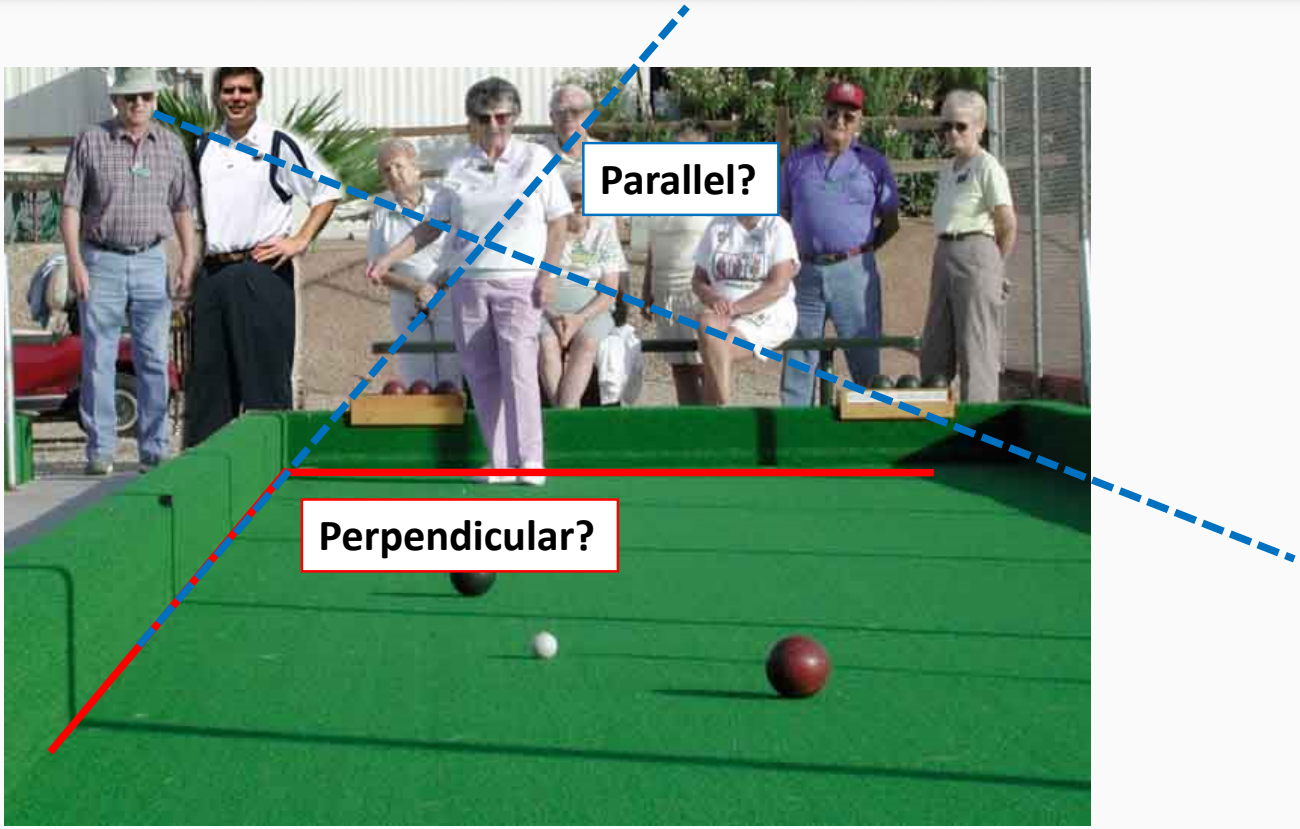
Projective geometry - what is lost?



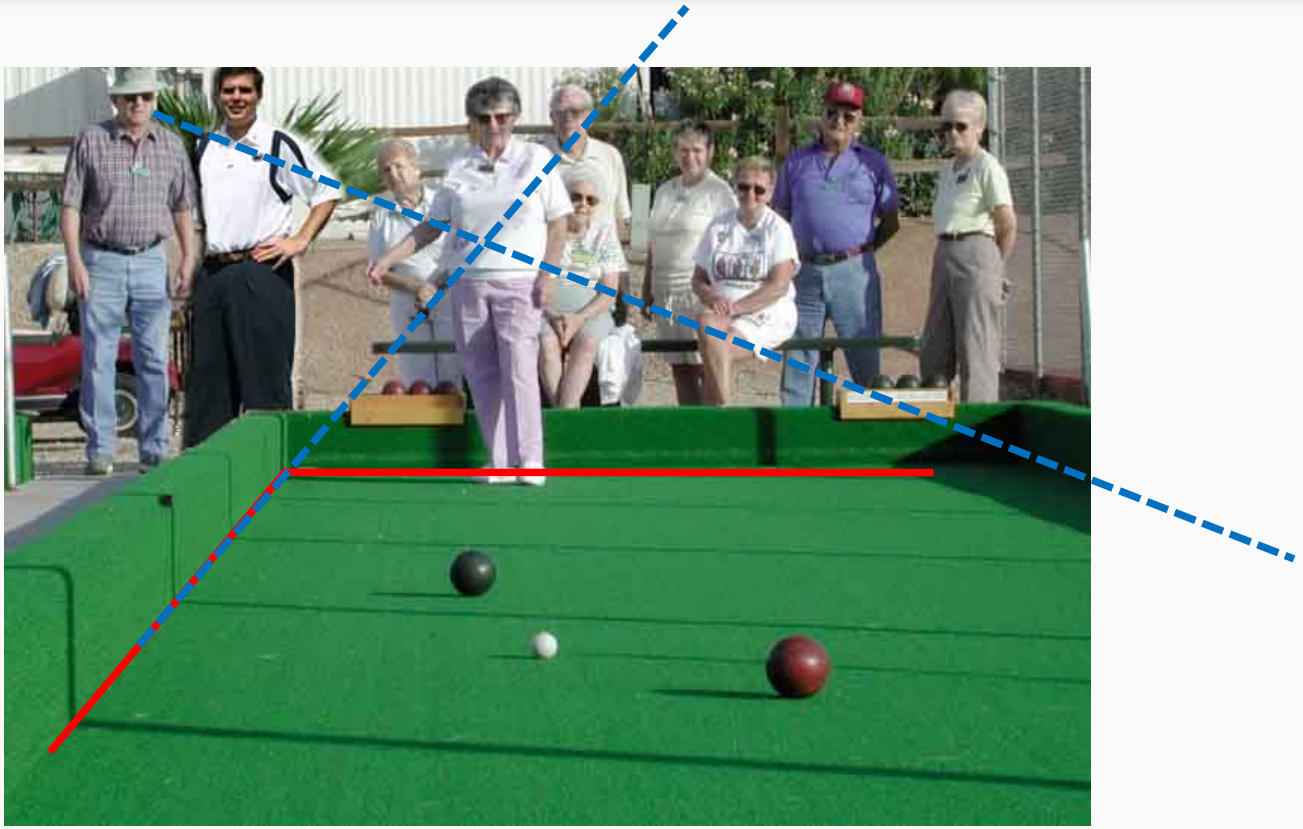
Length is not preserved



Angles are not preserved

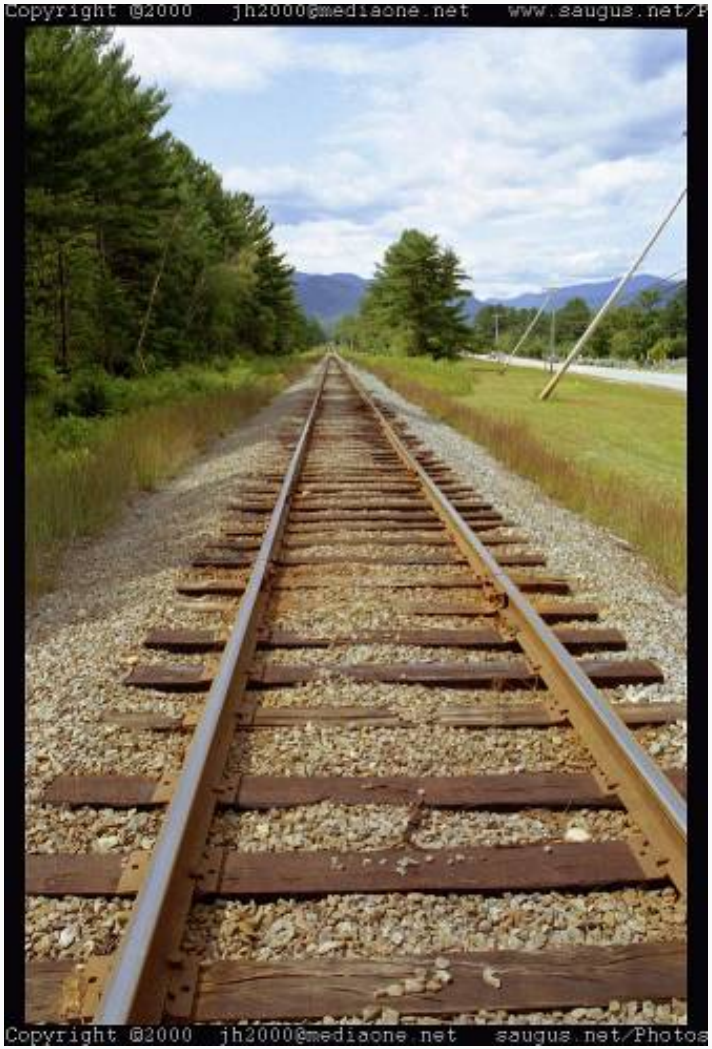


Straight lines are still straight

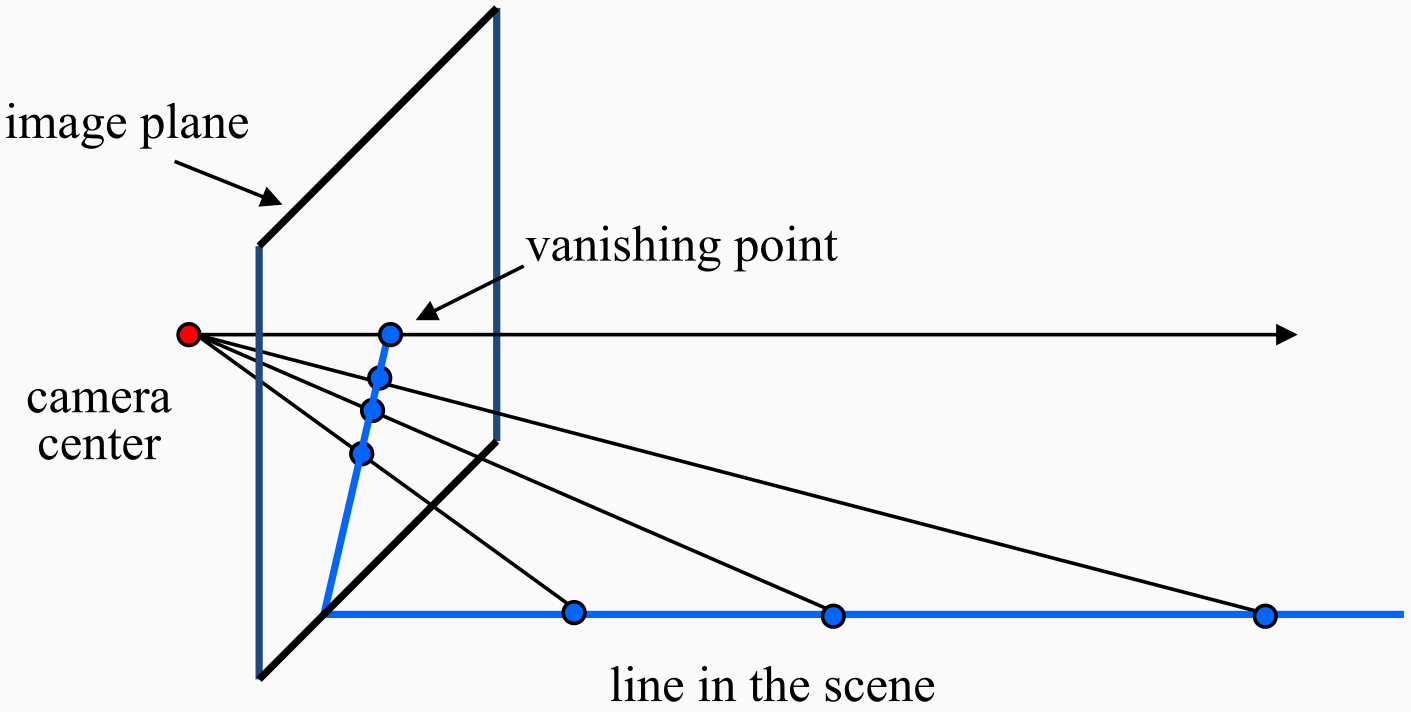


Vanishing points and lines

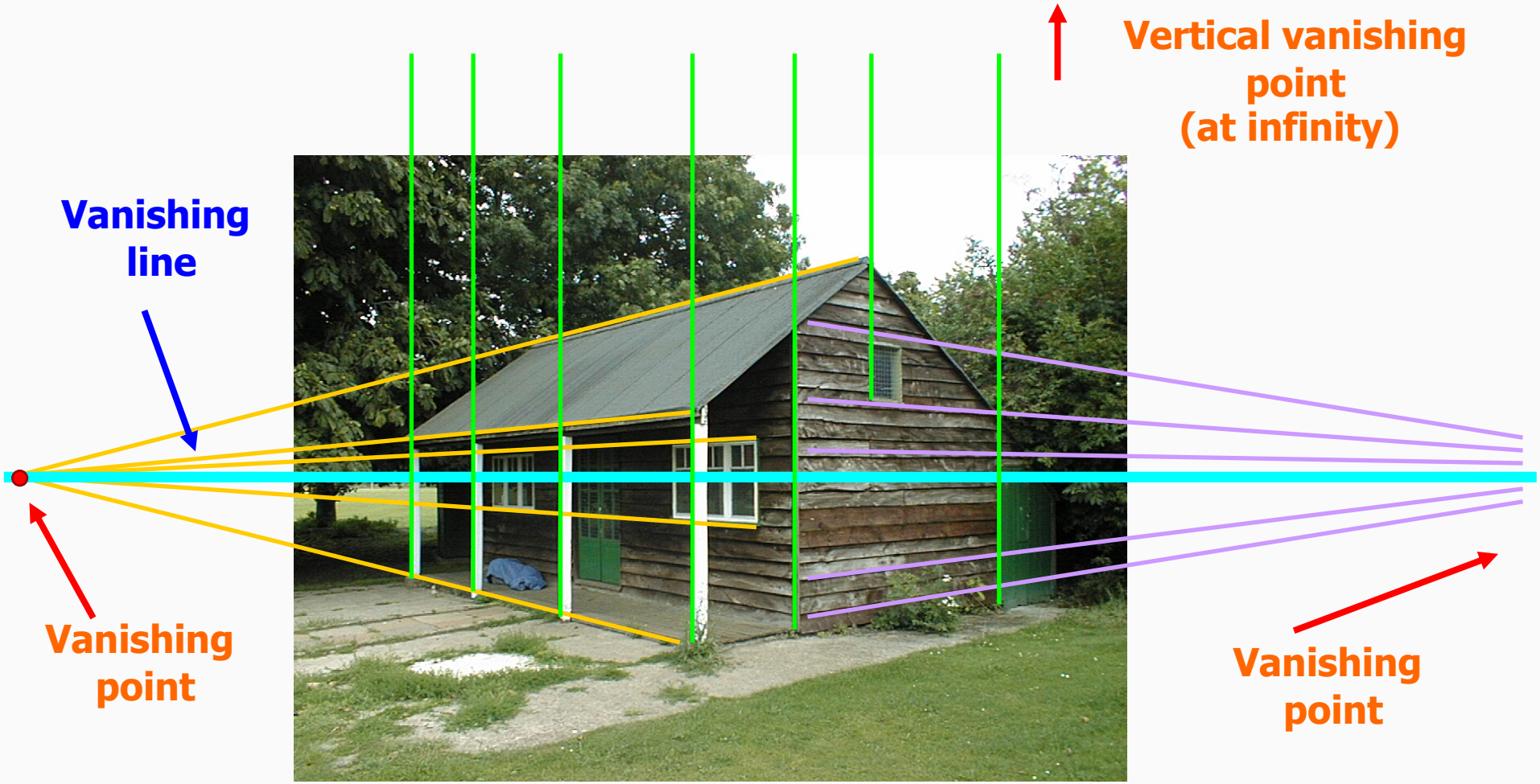
- Parallel lines in the world intersect at a “vanishing point”



Constructing the vanishing point of a line



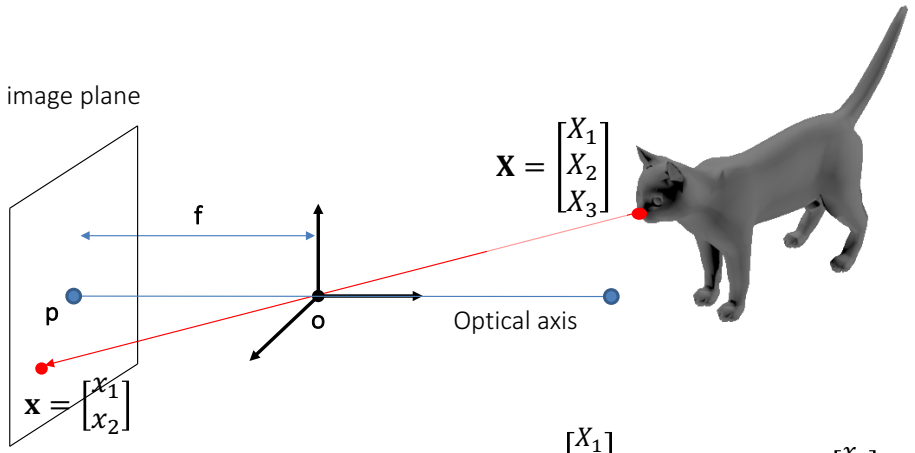
Vanishing points and lines



All parallel lines will have the same vanishing point.

Homogenous coordinates

- The projection $x = fX/x_3$ is non linear!
- Can be made linear using homogenous coordinates
- Homogenous coordinates allow for transforms to be concatenated easily



f – focal length
o – camera origin
p – principal point

The 3D point $\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}$ is imaged into $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ as:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} f\frac{X_1}{X_3} \\ f\frac{X_2}{X_3} \end{bmatrix}$$

Homogenous coordinates

Conversion to homogenous coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Conversion from homogenous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w) \qquad \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Invariance to scaling

$$k\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} kx \\ ky \\ kw \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{kx}{kw} \\ \frac{ky}{kw} \end{bmatrix} = \begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \end{bmatrix}$$

E.g. [1,2,3] is the same as [3,6,9] and both represent the **same** inhomogeneous point [0.33,0.66].

Basic geometry in homogenous coordinates

- Line equation: ax+by+c=0
- A pixel p in homogenous coordinates:
- Line is given by cross product of two points
- Intersection of two lines is given by cross product of the lines

$$line_i = \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix}$$

$$p_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}$$

$$line_{ij} = p_i \times p_j$$

$$q_{ij} = line_i \times line_j$$

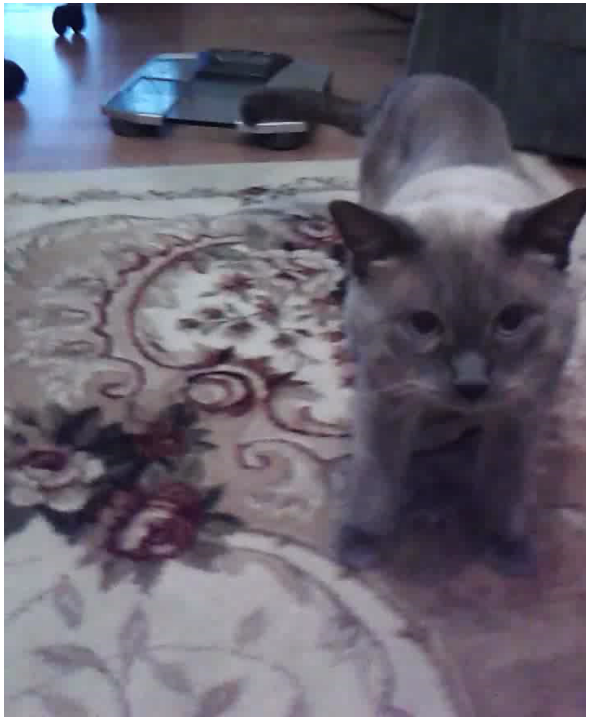
3D Euclidean transformation

- Cat moves through 3D space
- The movement of the nose can be described using an Euclidean Transform

$$\mathbf{X}'_{3\times1} = \boldsymbol{R}_{3\times3}\mathbf{X}_{3\times1} + \mathbf{t}_{3\times1}$$

$$\begin{pmatrix} \cos\beta\cos\gamma & -\sin\beta & \cos\beta \\ \sin\beta\cos\gamma & \cos\beta & -\sin\beta\sin\gamma \\ \sin\gamma & 0 & \cos\gamma \end{pmatrix}$$

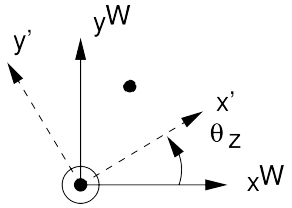
{ rotation } { translation }



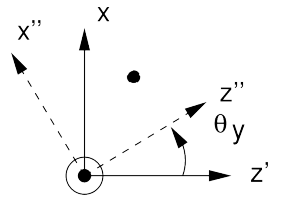
Building the 3D rotation matrix R

- R can be build from various representations (Euler ang., quaternion)
- Euler angles represent the rotation using three parameters, one for each axis

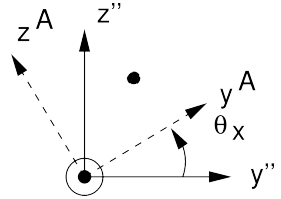
$$X' = R_z X^W = \begin{bmatrix} \cos\theta_z & \sin\theta_z & 0 \\ -\sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} X^W$$



$$X'' = R_y X' = \begin{bmatrix} \cos\theta_y & 0 & -\sin\theta_y \\ 0 & 1 & 0 \\ \sin\theta_y & 0 & \cos\theta_y \end{bmatrix} X'$$



$$X^A = R_x X'' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & \pm\sin\theta_x \\ 0 & \mp\sin\theta_x & \cos\theta_x \end{bmatrix} X''$$



$$\boldsymbol{R}^{CW} = \boldsymbol{R}_x \boldsymbol{R}_y \boldsymbol{R}_z$$

Order matters!

3D Euclidean transformation

- Concatenation of successive transforms is a mess!

$$\mathbf{X}_1 = \boldsymbol{R}_1\mathbf{X} + \mathbf{t}_1$$

$$\mathbf{X}_2 = \boldsymbol{R}_2\mathbf{X}_1 + \mathbf{t}_2$$

$$= \boldsymbol{R}_2(\boldsymbol{R}_1\mathbf{X} + \mathbf{t}_1) + \mathbf{t}_2 = (\boldsymbol{R}_2\boldsymbol{R}_1)\mathbf{X} + (\boldsymbol{R}_2\mathbf{t}_1 + \mathbf{t}_2)$$

Homogenous coordinates save the day!

- Replace 3D points $\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$ with homogenous versions $\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$

- The Euclidean transform becomes

$$\begin{bmatrix} \mathbf{X}' \\ 1 \end{bmatrix} = \boldsymbol{E} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix}$$

- Transformation can now be concatenated by matrix multiplication

$$\begin{bmatrix} \mathbf{X}_1 \\ 1 \end{bmatrix} = \boldsymbol{E}_{10} \begin{bmatrix} \mathbf{X}_0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} \mathbf{X}_2 \\ 1 \end{bmatrix} = \boldsymbol{E}_{21} \begin{bmatrix} \mathbf{X}_1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{X}_2 \\ 1 \end{bmatrix} = \boldsymbol{E}_{21} \boldsymbol{E}_{10} \begin{bmatrix} \mathbf{X}_0 \\ 1 \end{bmatrix}$$

More 3D-3D and 2D-2D transformations

Projective (15 dof):

$$\begin{bmatrix} X'_1 \\ X'_2 \\ X'_3 \\ X'_4 \end{bmatrix} = [\boldsymbol{P}_{4\times4}] \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

Affine (12 dof):

$$\begin{bmatrix} \mathbf{X}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{3\times3} & \mathbf{t}_3 \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix}$$

Similarity (7 dof):

$$\begin{bmatrix} \mathbf{X}' \\ 1 \end{bmatrix} = \begin{bmatrix} S\boldsymbol{R}_{3\times3} & \mathbf{t}_3 \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix}$$

Euclidean (6 dof):

$$\begin{bmatrix} \mathbf{X}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{3\times3} & \mathbf{t}_3 \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix}$$

Projective (aka Homography, 8 dof):

$$\begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} = [H_{3\times3}] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Affine (6 dof):

$$\begin{bmatrix} \mathbf{x}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{2\times2} & \mathbf{t}_2 \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}$$

Similarity (4 dof):

$$\begin{bmatrix} \mathbf{x}' \\ 1 \end{bmatrix} = \begin{bmatrix} S\boldsymbol{R}_{2\times2} & \mathbf{t}_2 \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}$$

Euclidean (3 dof):

$$\begin{bmatrix} \mathbf{x}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{2\times2} & \mathbf{t}_2 \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}$$

Examples of 2D-2D transforms

$$\begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Euclidean
3 DoF

$$\begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Similarity
4 DoF

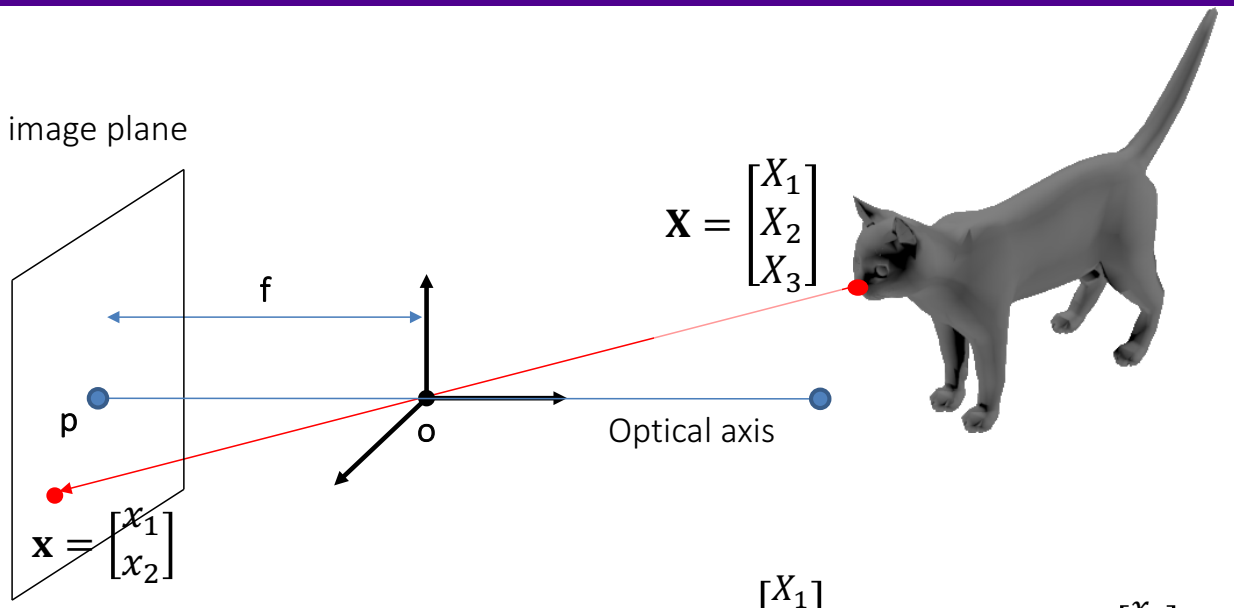
$$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Affine
6 DoF

$$\begin{bmatrix} h_{11} & h_{12} & h_{12} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

Projective
8 DoF

Perspective transformation (3D-2D)

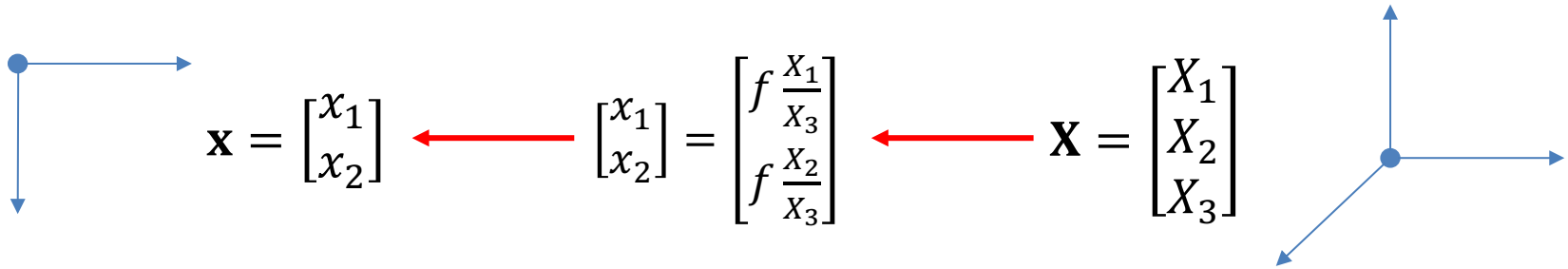


f – focal length
o – camera origin
p – principal point

The 3D point $\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}$ is imaged into $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ as:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} f\frac{X_1}{X_3} \\ f\frac{X_2}{X_3} \end{bmatrix}$$

Perspective using homogenous coordinates



$$\lambda \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \rightarrow \begin{matrix} \lambda x_1 = f X_1 \\ \lambda x_2 = f X_2 \\ \lambda = X_3 \end{matrix} \rightarrow \begin{matrix} x_1 = f\frac{X_1}{X_3} \\ \\ x_2 = f\frac{X_2}{X_3} \end{matrix}$$

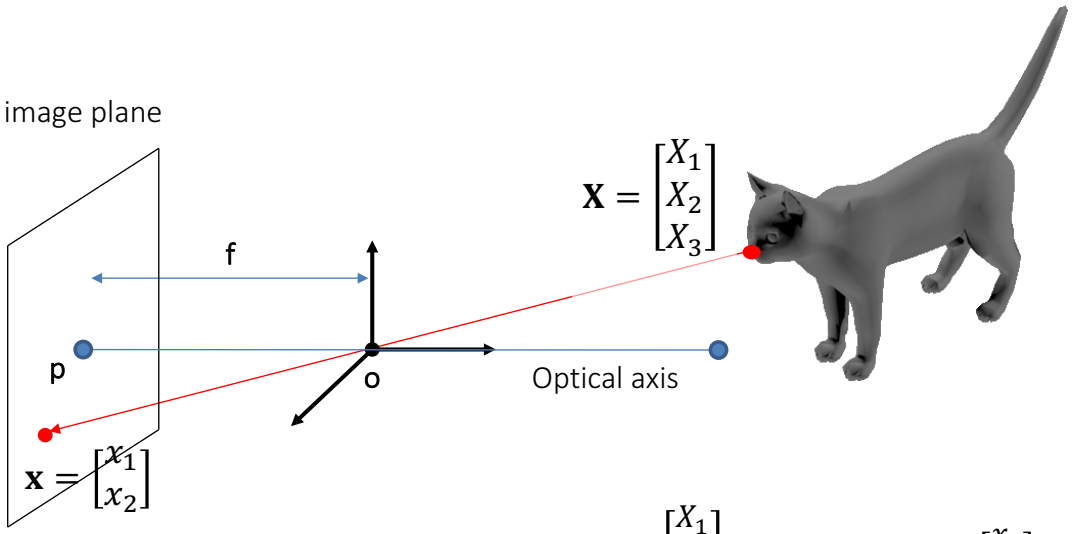
Perspective using homogenous coordinates

Image Point	Projection Matrix	World Point

$$\lambda \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix}$$

Wait! Our setup has several assumptions

- Camera at world origin
- Camera aligned with world coordinates
- Ideal pinhole camera



f – focal length
o – camera origin
p – principal point

The 3D point $\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}$ is imaged into $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ as:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} f\frac{X_1}{X_3} \\ f\frac{X_2}{X_3} \end{bmatrix}$$

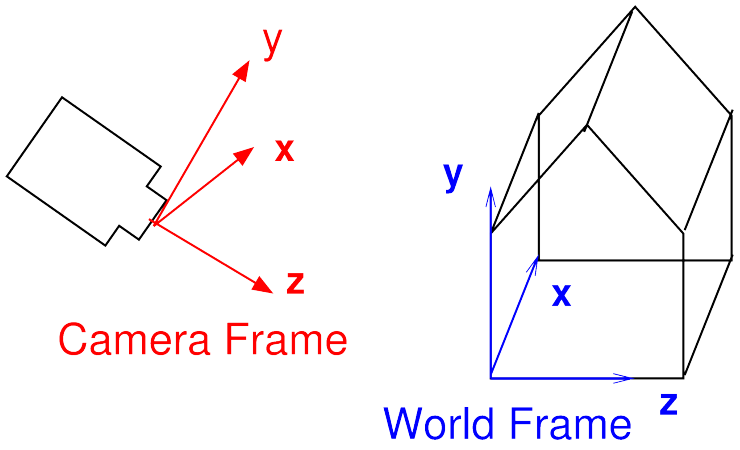
Removing the initial assumptions

- It is useful to split the overall projection matrix into three parts:
  - A part that depends on the internals of the camera (intrinsic)
  - A vanilla projection matrix
  - An Euclidean transformation between the world and camera frames (extrinsic)
- Assume first that the world is aligned with camera coordinates
  -> the extrinsic camera matrix is an identity

Image Point	Camera's Intrinsic Calibration	Projection matrix (vanilla)	Camera's Extrinsic Calibration	World Point
$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$\begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\mathbf{X}$

More realistic setting - camera pose

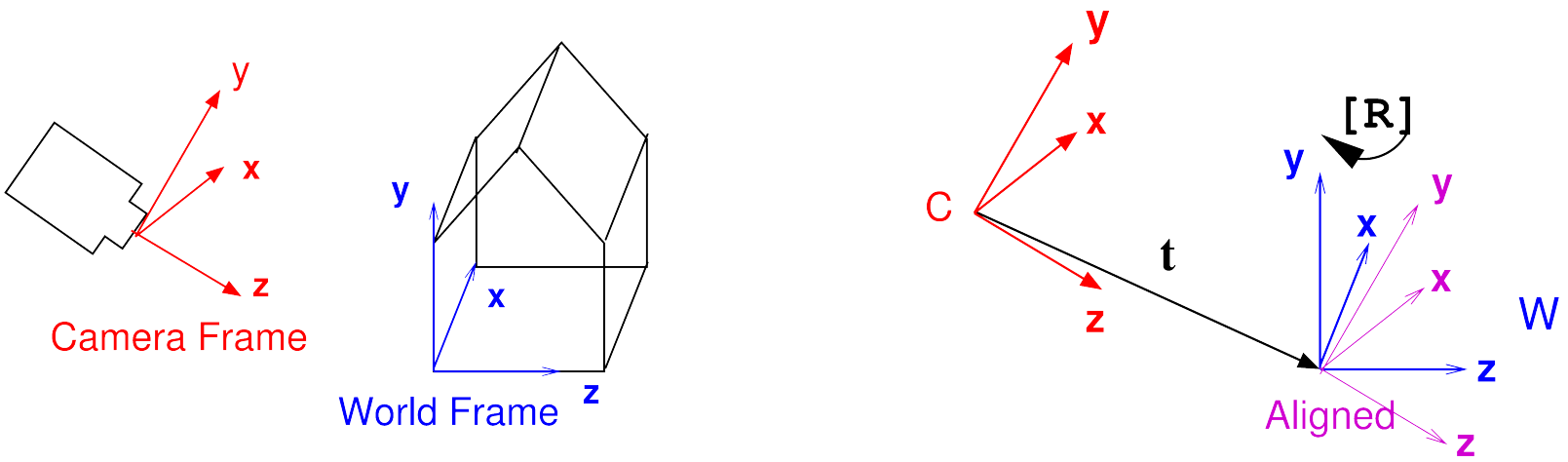
- Assume the camera is translated and rotated with respect to the world



The camera pose

- The non-ideal camera pose can be taken into account by first rotating and translating points from world frame to the camera frame

$$\begin{bmatrix} \mathbf{X}^{\mathbf{C}} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X}^{W} \\ 1 \end{bmatrix}$$



The intrinsic parameters

- Transformation to pixel units from metric units
- Describe the hardware properties of a real camera
  - The image plane might be skewed
  - The pixels might not be square

$$K = \begin{bmatrix} f & 0 & 0 \\ 0 & \gamma f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \frac{u_0}{f} \\ 0 & 1 & \frac{v_0}{\gamma f} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & s & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f & sf & u_o \\ 0 & \gamma f & v_o \\ 0 & 0 & 1 \end{bmatrix}$$

different scaling on x and y
γ is the aspect ratio.

Origin offset, (u_o, v_o) is the principal point.

s accounts for skew

Summary of steps from scene to image

- Move the scene point $(X^w,1)^T$ into camera coordinate system by 4x4 (extrinsic) Euclidean transformation:

$$\begin{bmatrix} \mathbf{X}^C \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X}^W \\ 1 \end{bmatrix}$$

- Project into ideal camera via the vanilla perspective transformation

$$\begin{bmatrix} \mathbf{x}' \\ 1 \end{bmatrix} = [\boldsymbol{I}|\mathbf{0}] \begin{bmatrix} \mathbf{X}^C \\ 1 \end{bmatrix}$$

- Map the ideal image into the real image using intrinsic matrix

$$\begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = \boldsymbol{K} \begin{bmatrix} \mathbf{x}' \\ 1 \end{bmatrix}$$

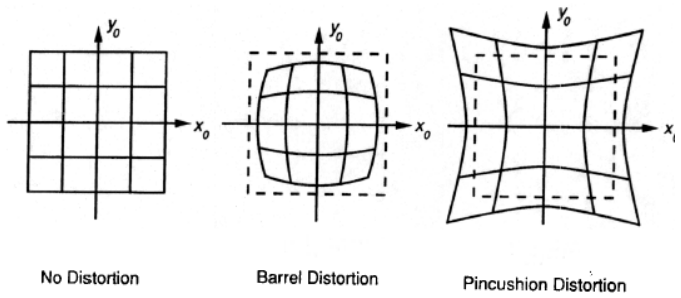
Camera projection matrix P

Image Point	Camera's Intrinsic Calibration	Projection matrix (vanilla)	Camera's Extrinsic Calibration	World Point
$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$\begin{bmatrix} f & sf & u_0 \\ 0 & \gamma f & v_o \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\mathbf{X}$

$$\mathbf{P}_{3\times4} = \boldsymbol{K}[\boldsymbol{R}|\mathbf{t}]$$

Beyond pinholes: Radial distortion

- Common in wide-angle lenses
- Creates non-linear terms in projection
- Usually handled by solving non-linear terms and then correcting the image

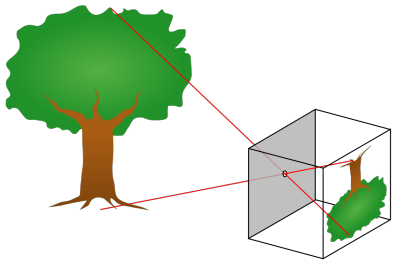


Original

Corrected

Things to remember

- Pinhole camera model



- Homogenous coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Camera projection matrix

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

The end