Robot Perception

Single View Geometry

Dr. Chen Feng
cfeng@nyu.edu

ROB-GY 6203, Fall 2022

Overview

+ AprilTags

* Homography Estimation

++ Ax=b, Ax=0

* Camera Calibration, Zhang's method

+ DLT

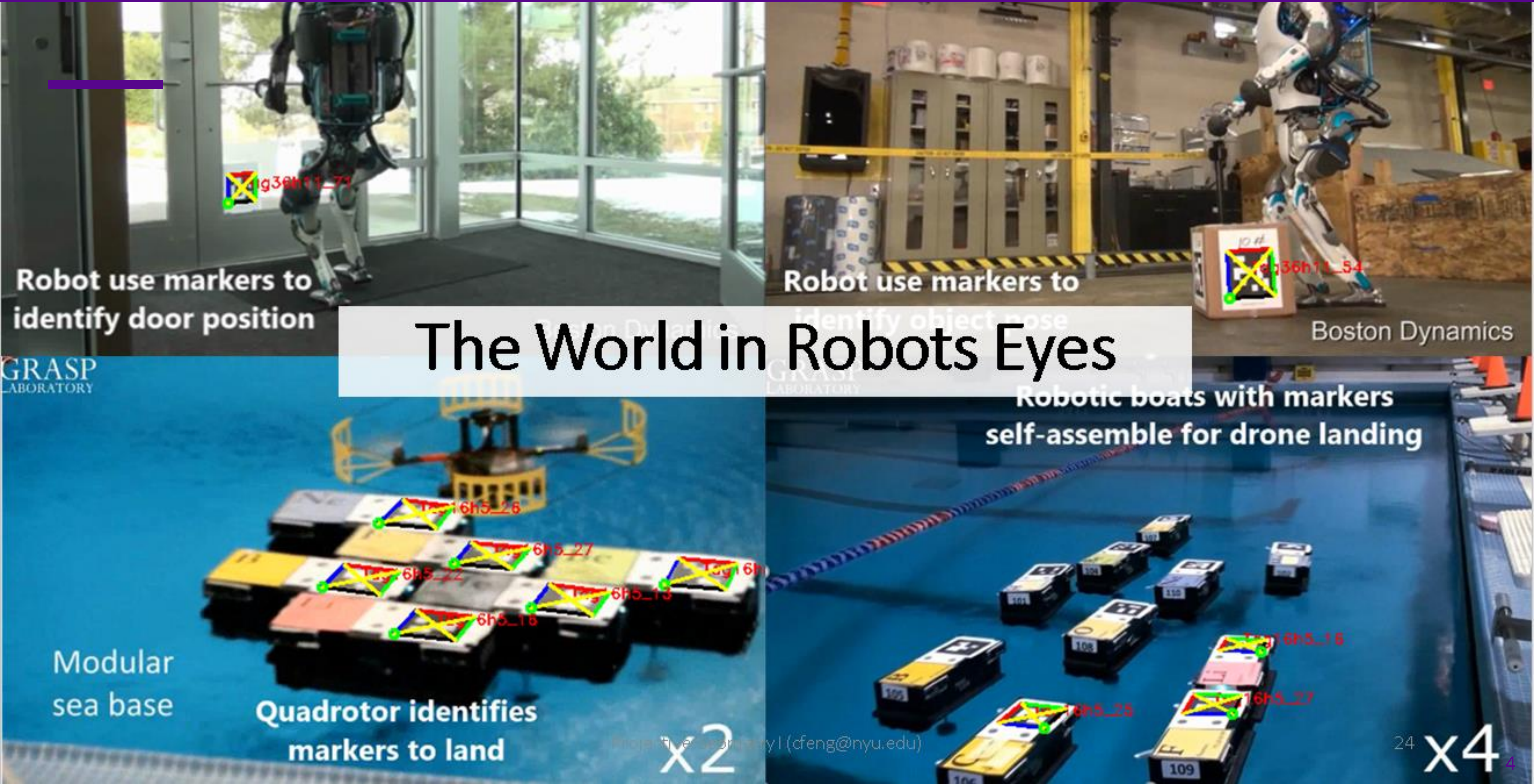
+ Vanishing Points & Lines

*: know how to code
++: know how to derive
+: know the concept

References

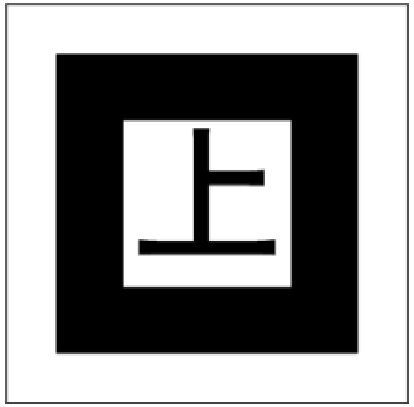
- HZ2003:
  - Section 2.3, 4.1, 4.4, 7.1, 7.2, 7.4, 8.6
- FP2011:
  - Section 1.2, 1.3, 12.1
- Sz2022:
  - Section 11.1, 11.4.5
- Co2011:
  - Section 11.2, 11.1
- Linear algebra:
  - Sz2011: section A.1.1, A.1.2, A.1.3, A.2, A.2.1
  - HZ2003: A5.1, A5.2, A5.3

The World in Robots Eyes

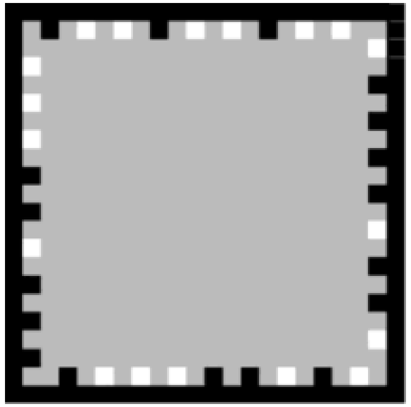


Fiducial Markers: More Than QR Codes

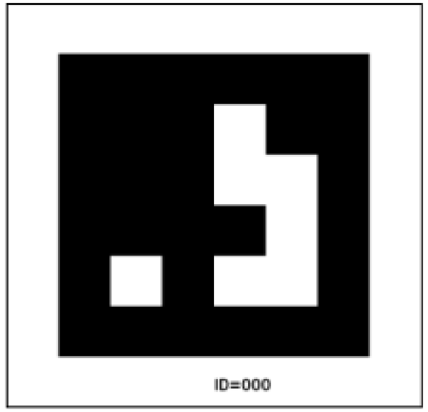
Fiducial:



(Kato and Billinghurst 1999)



(Wagner et al. 2008)



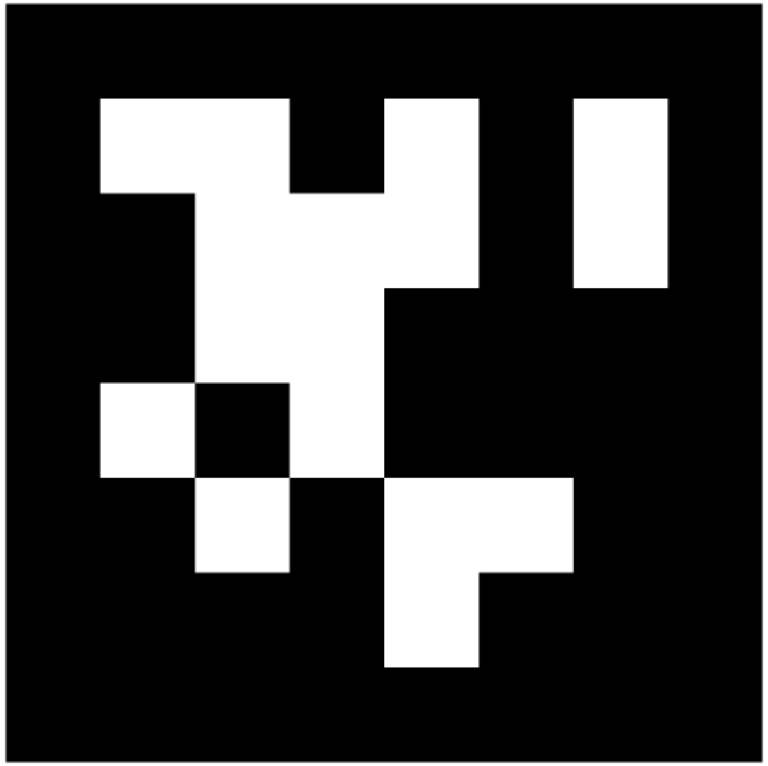
(Olson 2011)

Natural:
(Lepetit and Fua 2005)

https://developer.vuforia.com/library/articles/Solution/Natural-Features-and-Ratings

(Feng and Kamat 2013)

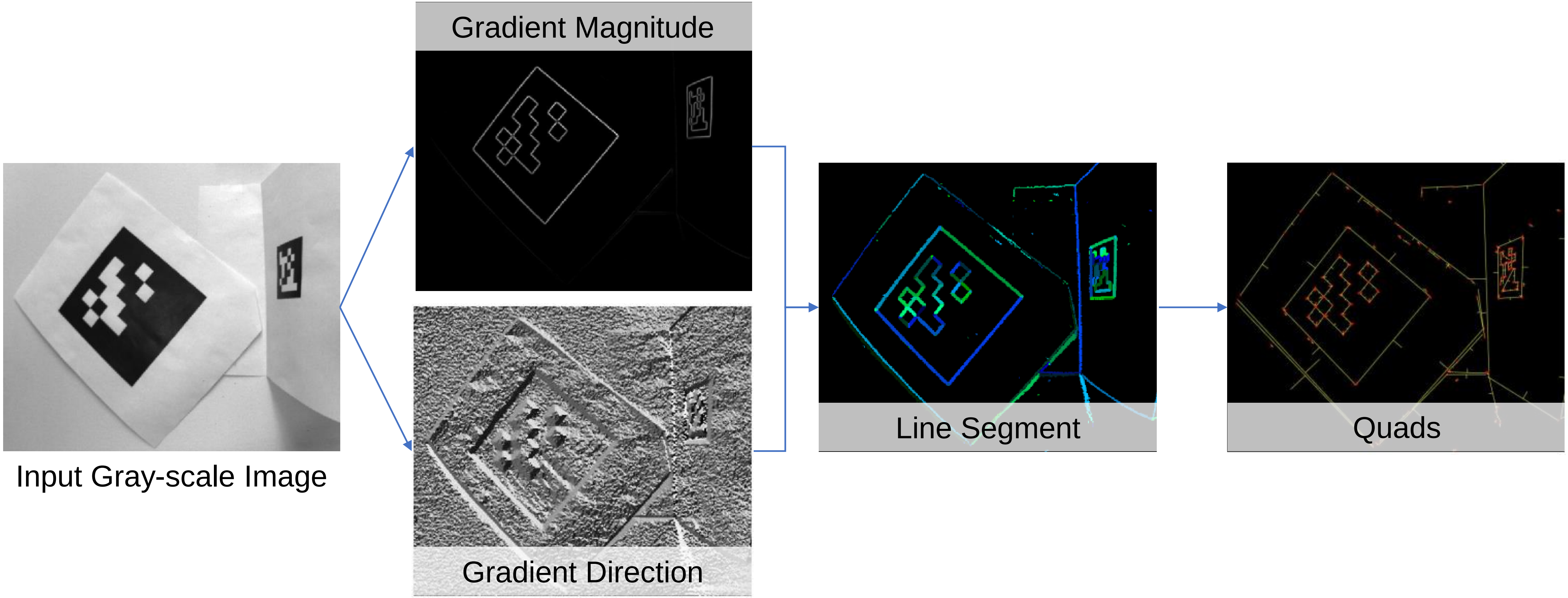
What Is an AprilTag?



$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \Rightarrow [110101011101011000101000010110000100] \Rightarrow \begin{cases} id = 0 \\ family = Tag36h11 \end{cases}$$

Olson 2011

How Is an AprilTag Detected?



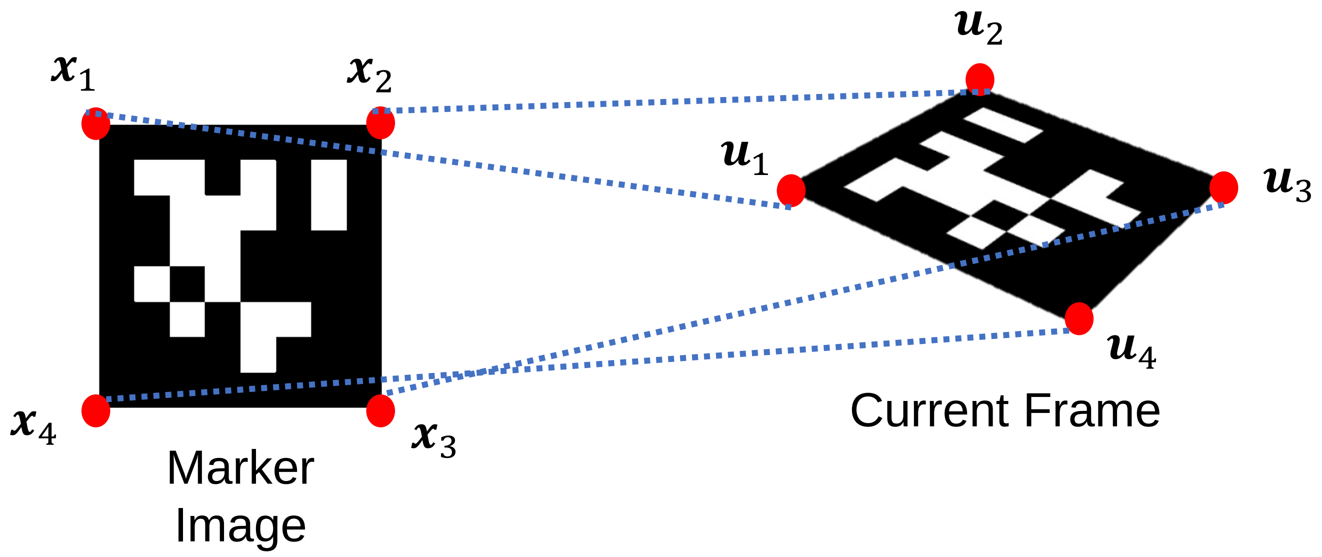
Olson 2011

Advantages of AprilTag

- Fast
  - >25Hz for 640x480 webcam Image on normal laptop
- Robust
  - Higher detection rate
  - Fewer false alarm
- Larger Range
  - Distance
  - View direction
  - Illumination

Max Detectable Distance (m)		Marker Angle (degree)			
		0	45	0	45
Marker Size (m^2)	0.2 x 0.2	6.10	4.88	11.28	8.84
	0.3 x 0.3	8.23	7.01	14.94	11.58
	0.46 x 0.46	13.41	11.28	25.91	21.64
	0.6 x 0.6	19.51	16.46	34.44	30.48
Image Resolution		640 x 480		1280 x 960	
Focal Length		850 pixels		1731 pixels	
Processing Rate		20 Hz		5 Hz	

AprilTags Provide Point Correspondences



- Useful for many projective geometry applications

Homography == Projective Transformation

Definition 2.9. A *projectivity* is an invertible mapping h from $\mathbb{P}^2$ to itself such that three points $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ lie on the same line if and only if $h(\mathbf{x}_1)$, $h(\mathbf{x}_2)$ and $h(\mathbf{x}_3)$ do.

- They all mean the same thing:
  - Homography
  - Projectivity
  - Collineation

Theorem 2.10. *A mapping $h : \mathbb{P}^2 \rightarrow \mathbb{P}^2$ is a projectivity if and only if there exists a non-singular 3×3 matrix $\mathtt{H}$ such that for any point in $\mathbb{P}^2$ represented by a vector $\mathbf{x}$ it is true that $h(\mathbf{x}) = \mathtt{H}\mathbf{x}$.*

Hartley & Zisserman 2011

Homography == Projective Transformation

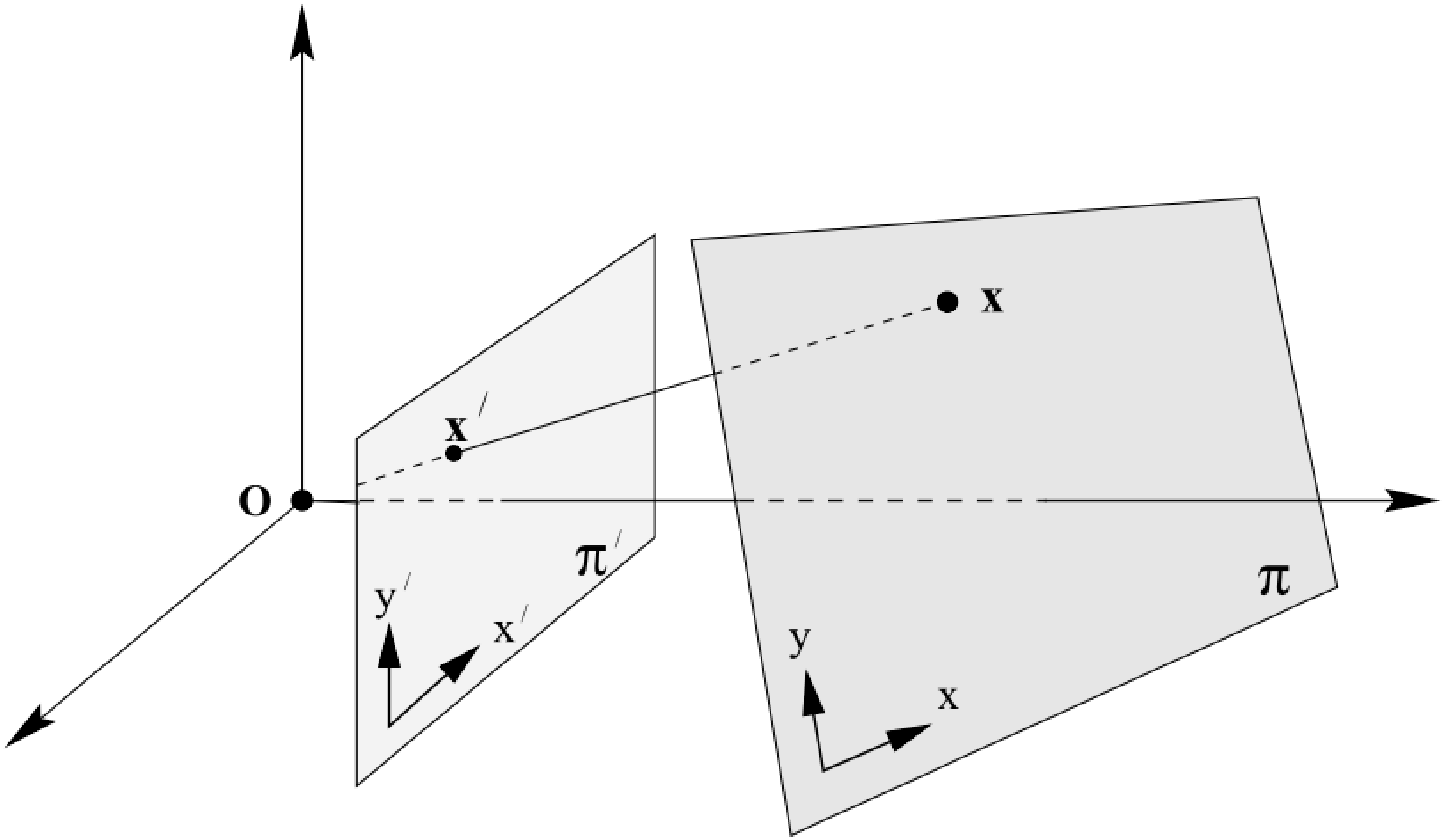
Definition 2.11. Projective transformation. A planar projective transformation is a linear transformation on homogeneous 3-vectors represented by a $\boxed{\text{non-singular}}$ 3×3 matrix:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \qquad (2.5)$$

or more briefly, $\mathbf{x}' = \mathtt{H}\mathbf{x}$.

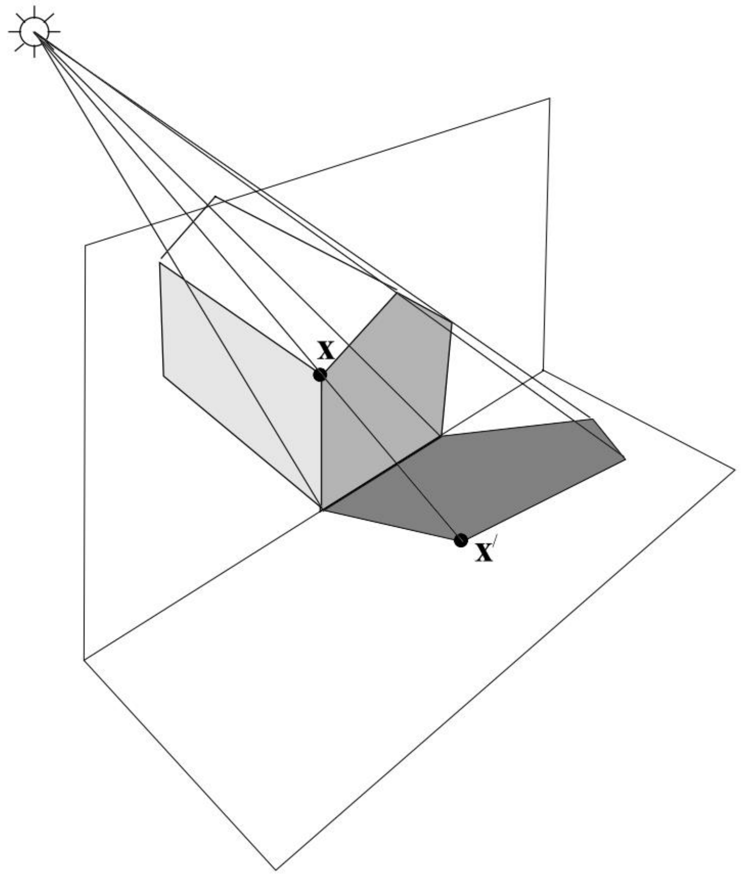
Hartley & Zisserman 2011

Perspective Transformation $\subset$ Homography



Hartley & Zisserman 2011

Real-life Example of Perspectivity

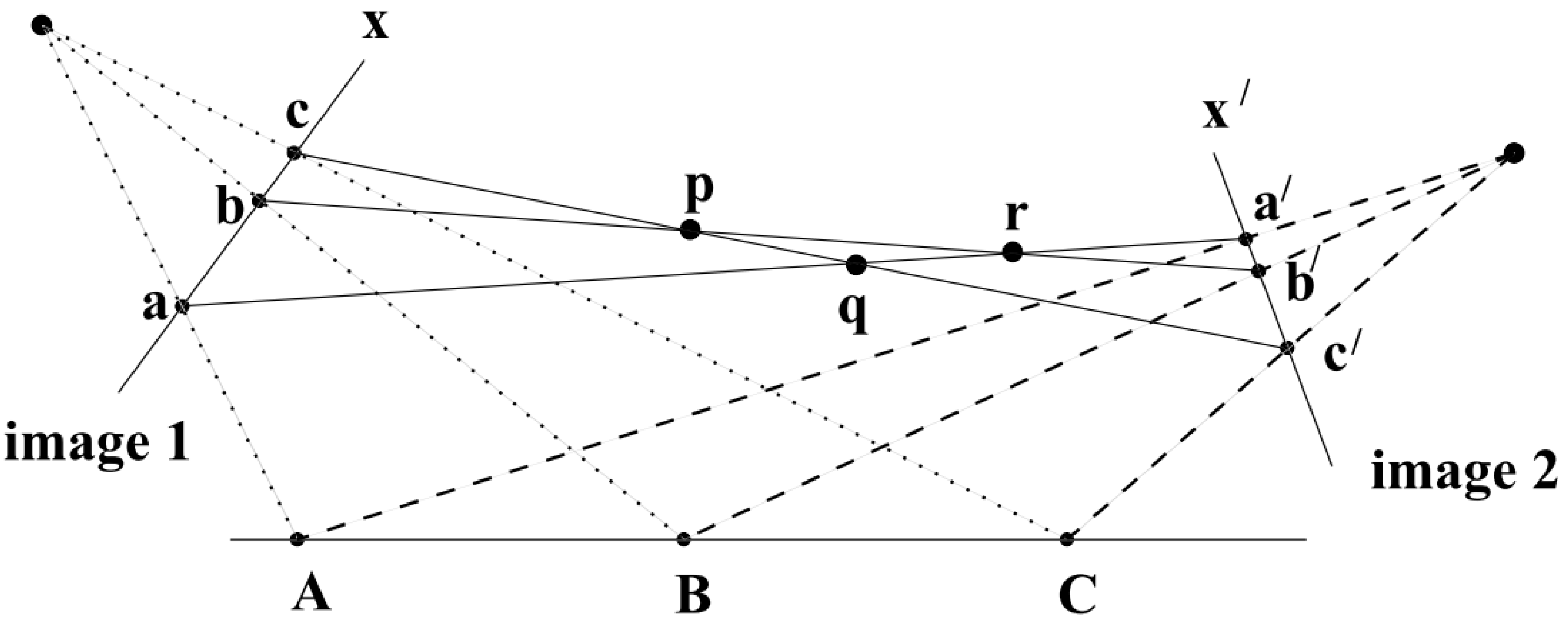


Hartley & Zisserman 2011

Homography vs. Perspectivity

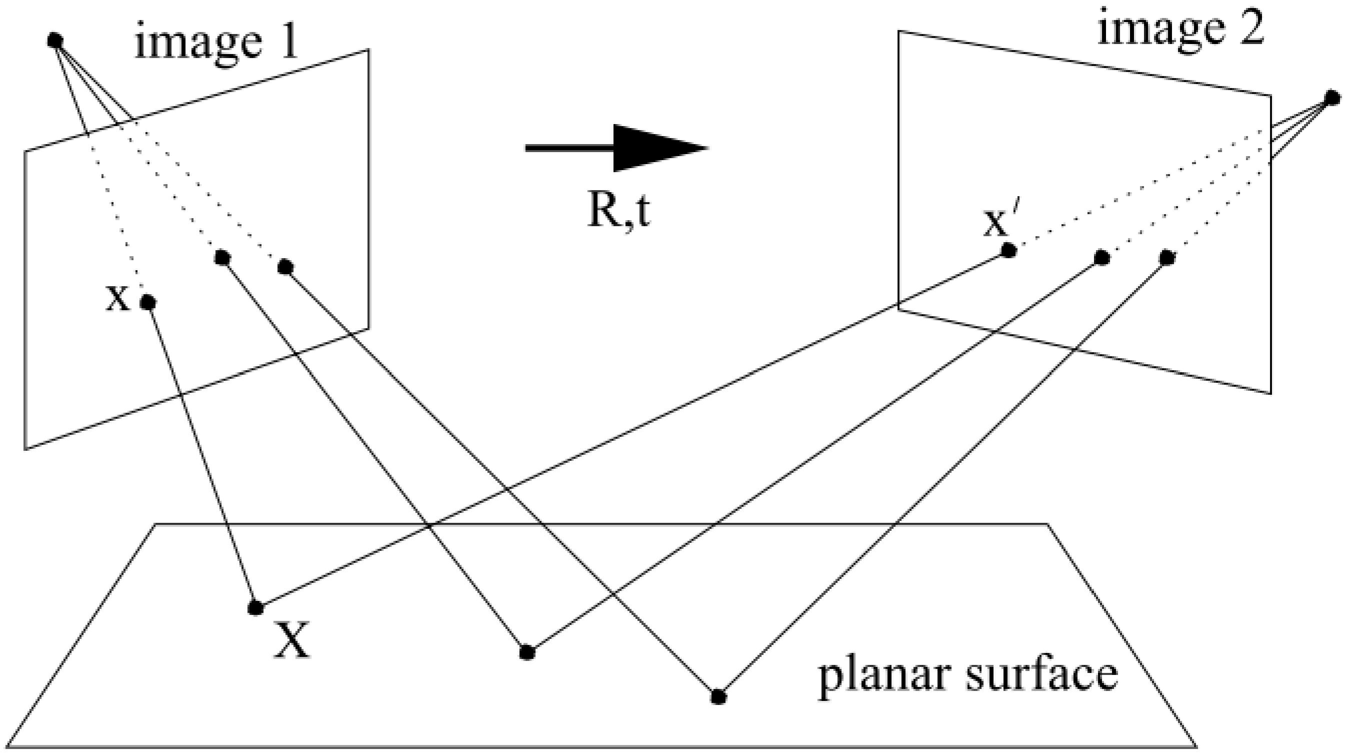
homography

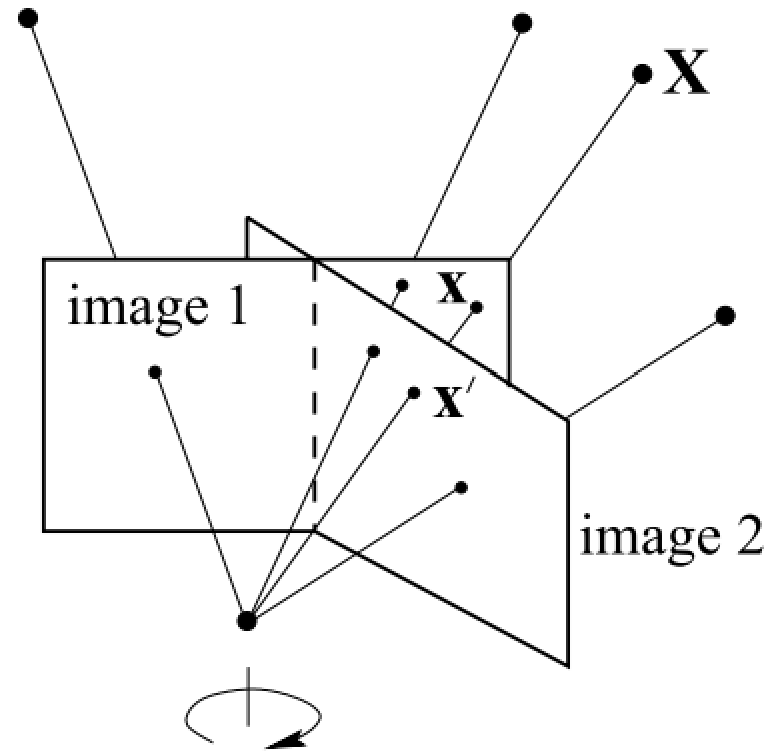
The composition of two (or more) perspectivities is a projectivity, but not, in general, a perspectivity.



Hartley & Zisserman 2011

Which Is a Non-Perspective Homography?





Hartley & Zisserman 2011

Application of Perspective Homography

Hartley & Zisserman 2011

Removing Perspective Distortion?

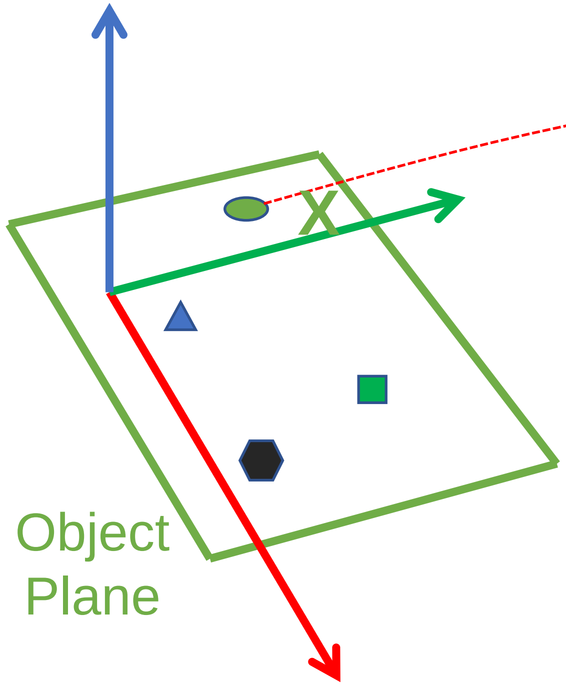
Hartley & Zisserman 2011

Removing Perspective Distortion?

Hartley & Zisserman 2011

How to Estimate a Homography?

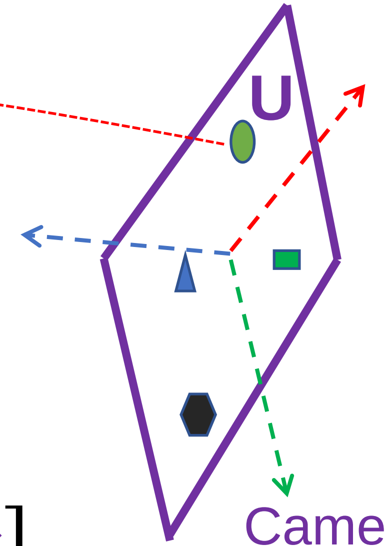
corresponding points



Object Plane

$$\mathbf{HX} \propto \mathbf{U}$$

$$\begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \propto \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$



Camera Plane (Image)

Estimating Homography: DLT

- 2D direct linear transformation (DLT) algorithm
- Find multiple $\mathbf{X} \leftrightarrow \mathbf{U}$ correspondences (≥ 4) between a planar object and an image
- Each correspondence leads to 2 independent linear equations with homography as unknown parameters:

$$u(h_7 x + h_8 y + h_9) = h_1 x + h_2 y + h_3$$
$$v(h_7 x + h_8 y + h_9) = h_4 x + h_5 y + h_6$$

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -w'_i \mathbf{x}_i^\mathsf{T} & y'_i \mathbf{x}_i^\mathsf{T} \\ w'_i \mathbf{x}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} & -x'_i \mathbf{x}_i^\mathsf{T} \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

- This leads to a homogeneous system of linear equations

$$xh_1 + yh_2 + h_3 - uxh_7 - uyh_8 - uh_9 = 0$$
$$xh_4 + yh_5 + h_6 - vxh_7 - vyh_8 - vh_9 = 0$$

$$\begin{bmatrix} x & y & 1 & & & & -ux & -uy & -u \\ & & & x & y & 1 & -vx & -vy & -v \end{bmatrix} [h_1\ h_2\ h_3\ h_4\ h_5\ h_6\ h_7\ h_8\ h_9]^T = 0$$

$$\mathbf{Ah} = \mathbf{0}$$

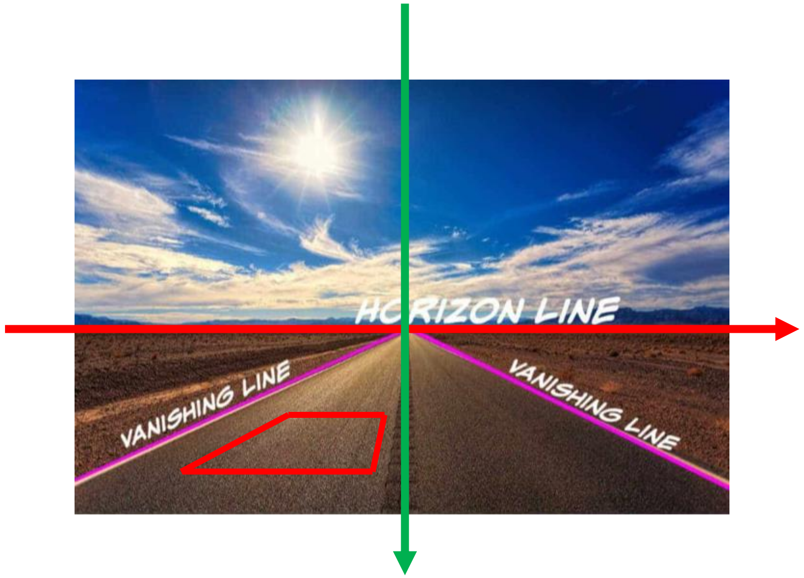
2D DLT Using Inhomogeneous Homography

- Set $h_9 = 1, \quad \widetilde{\boldsymbol{h}} = [h_1, h_2, \cdots, h_8]$

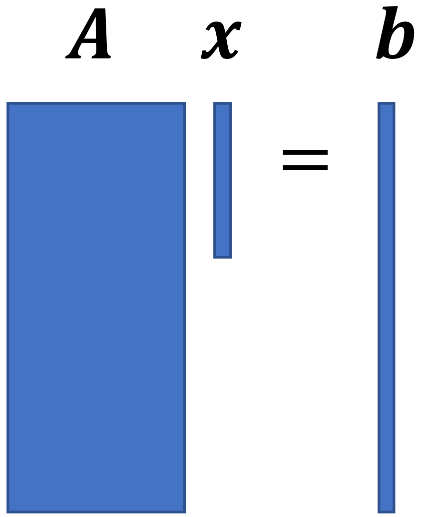
$$\begin{bmatrix} x & y & 1 & & & & -ux & -uy \\ & & & x & y & 1 & -vx & -vy \end{bmatrix} [h_1\ h_2\ h_3\ h_4\ h_5\ h_6\ h_7\ h_8]^T = \begin{bmatrix} u \\ v \end{bmatrix}$$

- Solve by least square: $\left(A^T A\right)^{-1} A^T b$

- Potential issues
  - What if $h_9 = 0$?
  - Does this happen often?

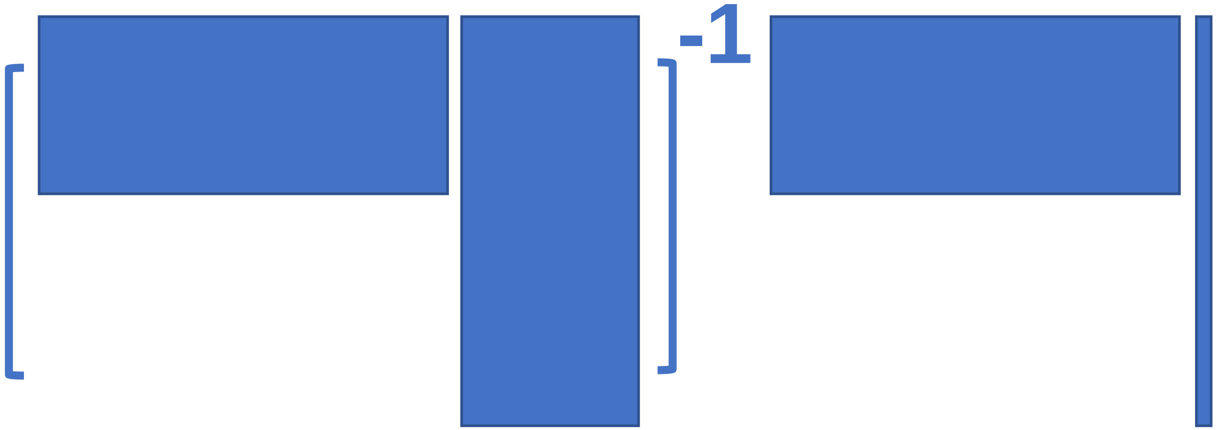


Solving Ax=b



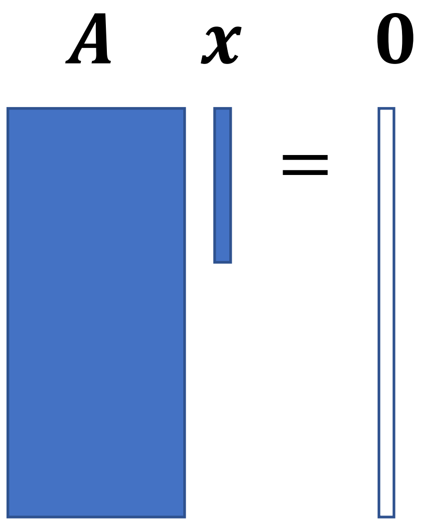
- A: design matrix
  - shape: m x n
  - m>>n
  - Typically, full column-rank
- x: unknowns
  - shape: n x 1
- b: observed data
  - shape: m x 1
- Solve by least squares: x* = inv(A’A)A’b
  - Solving normal equation: A’Ax=A’b
  - Least squares residual:
    - Observed - Predicted
    - b - Ax*
  - Residual is usually not zero in real world problems!

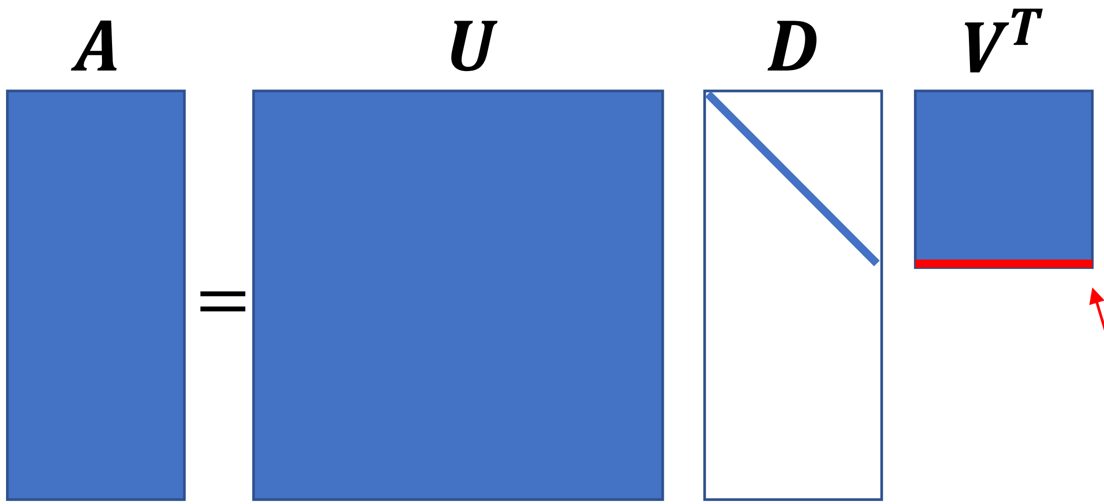
Least squares solution $\boldsymbol{x}^*$



Solving Ax=0

- A: data matrix
  - shape: m x n
  - m >> n
  - rank(A) = n when data contains noise: full column-rank
- x: unknowns
  - shape: n x 1
- Seek for an approximation instead of exact non-trivial solutions
- Add a constraint: ||x||=1
- Solve by SVD: A=UDV’
  - x*=last column of V, if diag(D) is descending order
  - diag(D): non-negative
    - called singular values





Least squares solution x^*

Why Ax=0 can be solved by A=UDV’ ?

- Problem conversion 1:
  - min ||Ax||, s.t. ||x||=1
- ||UDV’x|| == ||DV’x||
- ||x|| == ||V’x||
- Problem conversion 2:
  - min ||DV’x||, s.t. ||V’x||=1
- Change of variable: y=V’x
  - min ||Dy||, s.t. ||y||=1

Why $\boldsymbol{y}^*$ should be (0,0,…,0,1)’

$$\|\boldsymbol{D}\boldsymbol{y}\|^2 = \boldsymbol{y}^T\boldsymbol{D}^T\boldsymbol{D}\boldsymbol{y} = \sum_i \sigma_i^2 y_i^2$$
$$= \sigma_1^2 y_1^2 + \sigma_2^2 y_2^2 + \cdots + \sigma_n^2 y_n^2$$
$$= \sigma_1^2 y_1^2 + \sigma_2^2 y_2^2 + \cdots + \sigma_n^2 \left(1 - y_1^2 - y_2^2 \cdots - y_{n-1}^2\right)$$
$$= \left(\sigma_1^2 - \sigma_n^2\right) y_1^2 + \left(\sigma_2^2 - \sigma_n^2\right) y_2^2 + \cdots + \left(\sigma_{n-1}^2 - \sigma_n^2\right) y_{n-1}^2 + \sigma_n^2$$
$$\geq \sigma_n^2 \ (= only\ when\ y_1 = y_2 = \cdots = y_{n-1} = 0\ and\ y_n = 1)$$

- D is diagonally descending! => y*=(0,0,…,0,1)’ => x*=last column of V

Solving 2D DLT Using SVD

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix $\mathtt{H}$ such that $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

Algorithm

$$\begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

(i) For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ compute the matrix $\mathtt{A}_i$ from (4.1). Only the first two rows need be used in general.

(ii) Assemble the n 2×9 matrices $\mathtt{A}_i$ into a single $2n \times 9$ matrix $\mathtt{A}$.

(iii) Obtain the SVD of $\mathtt{A}$ (section A4.4($p585$)). The unit singular vector corresponding to the smallest singular value is the solution $\mathbf{h}$. Specifically, if $\mathtt{A} = \mathtt{UDV}^\top$ with $\mathtt{D}$ diagonal with positive diagonal entries, arranged in descending order down the diagonal, then $\mathbf{h}$ is the last column of $\mathtt{V}$.

(iv) The matrix $\mathtt{H}$ is determined from $\mathbf{h}$ as in (4.2).

Hartley & Zisserman 2011

Another Practical Issue

- Numerical issues
  - A typical image point: (100,100,1)
  - xx': 10^4
  - xw': 10^2
  - ww': 1

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -w_i'\mathbf{x}_i^\mathsf{T} & y_i'\mathbf{x}_i^\mathsf{T} \\ w_i'\mathbf{x}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} & -x_i'\mathbf{x}_i^\mathsf{T} \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

- Data normalization is essential for DLT!
  - All entries in A should have similar magnitude
  - Pre-process your data using similarity transformation
    - Zero-mean (de-mean)
    - Unit-variance

A Complete 2D DLT Algorithm

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix $\mathtt{H}$ such that $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

Algorithm

(i) **Normalization of $\mathbf{x}$:** Compute a similarity transformation $\mathtt{T}$, consisting of a translation and scaling, that takes points $\mathbf{x}_i$ to a new set of points $\tilde{\mathbf{x}}_i$ such that the centroid of the points $\tilde{\mathbf{x}}_i$ is the coordinate origin $(0,0)^\mathsf{T}$, and their average distance from the origin is $\sqrt{2}$.

(ii) **Normalization of $\mathbf{x}'$:** Compute a similar transformation $\mathtt{T}'$ for the points in the second image, transforming points $\mathbf{x}'_i$ to $\tilde{\mathbf{x}}'_i$.

(iii) **DLT:** Apply algorithm 4.1(p91) to the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ to obtain a homography $\widetilde{\mathtt{H}}$.

(iv) **Denormalization:** Set $\mathtt{H} = \mathtt{T}'^{-1}\widetilde{\mathtt{H}}\mathtt{T}$.

Hartley & Zisserman 2011

Homography and Camera Pose

- Using perspective projection equation:

$$\mathbf{u} \propto \mathbf{K}(\mathbf{R}\mathbf{X} + \mathbf{t})$$

- World point $\mathbf{X}$ lies on a plane, lets call it plane Z=0:

$$\mathbf{X} = \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$$

- Write rotation matrix as:

$$\mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3]$$

- Plug into the first equation:

$$\mathbf{u} \propto \mathbf{K}\left([\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3] \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} + \mathbf{t} \right) \equiv \mathbf{u} \propto \underbrace{\mathbf{K}[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}]}_{\mathbf{H}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Single-view homography decomposition

- $\mathbf{H} \propto \mathbf{K}[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}]$

- Assume we calibrated the camera, so **K** is known to us
- $\mathbf{K}^{-1}\mathbf{H} \propto [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}]$

- If let

$$\mathbf{K}^{-1}\mathbf{H} \stackrel{\text{def}}{=} [\boldsymbol{a}_1 \quad \boldsymbol{a}_2 \quad \boldsymbol{a}_3]$$

- Translation

$$\mathbf{t} = \boldsymbol{a}_3 / \|\boldsymbol{a}_1\|$$

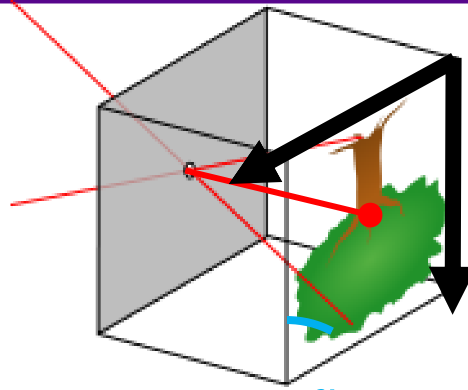
- Rotation

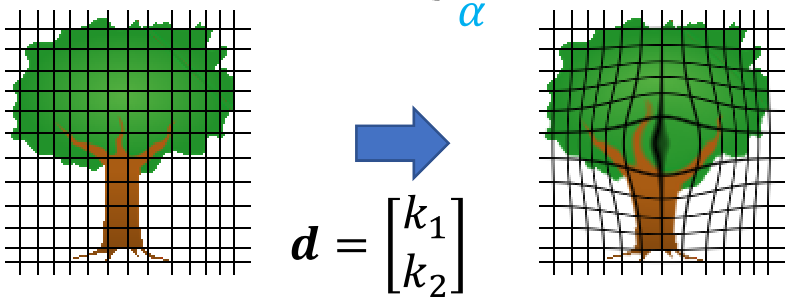
$$\mathbf{R} = [\boldsymbol{a}_1/\|\boldsymbol{a}_1\| \quad \boldsymbol{a}_2/\|\boldsymbol{a}_1\| \quad \boldsymbol{a}_1 \times \boldsymbol{a}_2/\|\boldsymbol{a}_1\|^2]$$

Camera Calibration

- To find out intrinsic parameters of a camera
  - **Linear:** $\mathbf{K} = ?$
  - **Non-linear:** $\mathbf{d} = ?$
- Intrinsic parameter values are generally static
  - Only need to be calibrated once
  - Unless the camera has been shipped for long distances
- Why?
  - Reduce uncertainties/unknowns in the projection system
  - Improve accuracy
- How?
  - $\mathbf{K}$ and $\mathbf{d}$ can not be easily measured directly
  - Has to be solved using perspective projection equation indirectly

$$\mathbf{K} = \begin{bmatrix} f_x & \alpha & c_x \\ & f_y & c_y \\ & & 1 \end{bmatrix}$$





Calibration with a 3D Rig - Simple

1. Form a 3D structure (rig) with multiple markers
2. Precisely measure each marker's corner point 3D positions ($\mathbf{X}$) in a same coordinate frame
3. Take an image of the 3D structure
4. Solve camera matrix using the 3D Direct Linear Transformation (**DLT**) algorithm
5. Decompose camera matrix to get camera intrinsics

3D DLT for Computing Camera Matrix

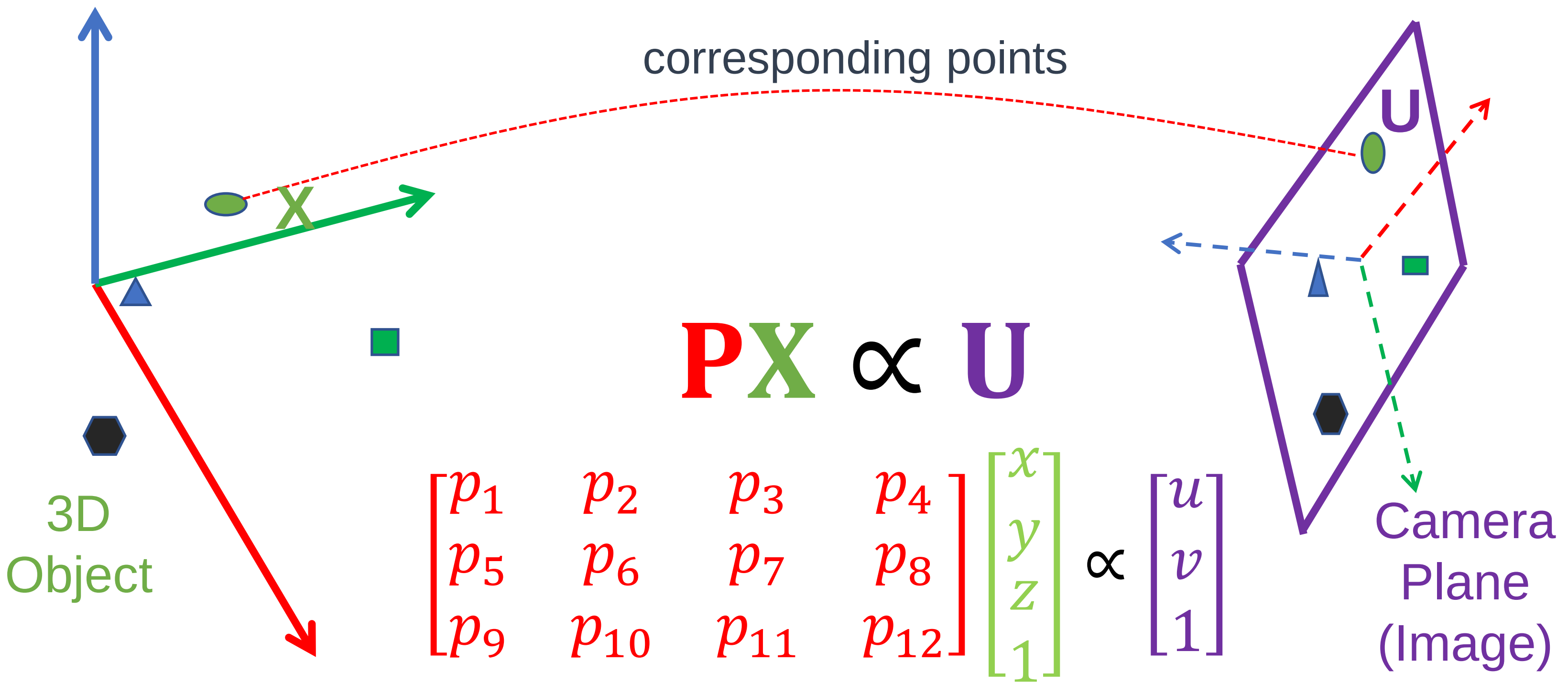
- Recall what is camera matrix:

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$P = K[R \mid t]$$

3D DLT for Computing Camera Matrix

- Similar to 2D homography



$$\mathbf{P}\mathbf{X} \propto \mathbf{U}$$

$$\begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \propto \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

3D DLT for computing camera matrix

- Find multiple $\mathbf{X} \leftrightarrow \mathbf{U}$ correspondences (≥ 6) between a planar object and an image

- Each correspondence lead to 2 independent linear equations with homography as unknown parameters:

$$\begin{bmatrix} \mathbf{0}^\top & -w_i\mathbf{X}_i^\top & y_i\mathbf{X}_i^\top \\ w_i\mathbf{X}_i^\top & \mathbf{0}^\top & -x_i\mathbf{X}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \\ \mathbf{P}^3 \end{pmatrix} = \mathbf{0}$$

- This lead to a homogeneous system of linear equation

$$\mathbf{Ah} = \mathbf{0}$$

- Solve by performing Singular Value Decomposition on **A**

Decomposing Camera Matrix

$$\begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \quad \mathrm{P} = \mathrm{K}[\mathrm{R} \mid \mathbf{t}] \quad \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

RQ-decomposition

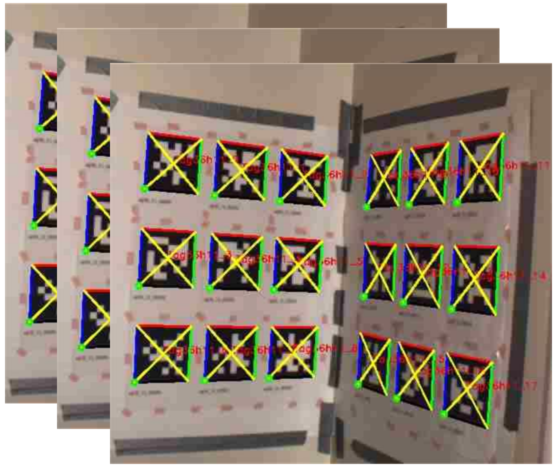
- Bonus: camera extrinsics!

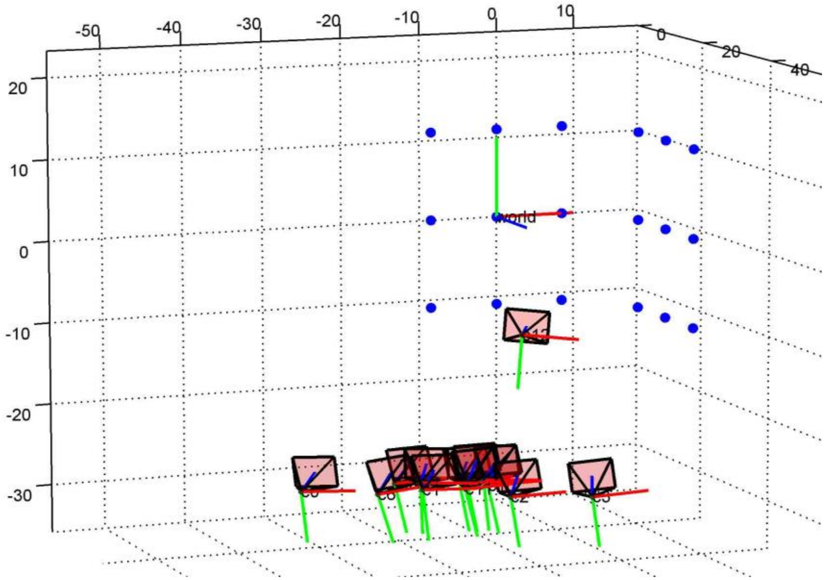
Calibration with a 3D Rig - Complete

1. Form a 3D structure (rig) with multiple (M) markers
2. Precisely measure each marker's corner point 3D positions ($\mathbf{X}$) in a same coordinate frame
3. Take N images of the 3D structure
4. Solve the **bundle adjustment** equation:

$$\underset{\mathbf{K},\{\mathbf{R}_i,\mathbf{t}_i\}}{\arg\min}\sum_{i=1}^{N}\sum_{j=1}^{M}\left\|\mathbf{U}_{i,j}-\mathbf{K}(\mathbf{R}_i\mathbf{X}_j+\mathbf{t}_i)\right\|^2$$

5. Initialization using the 3D Direct Linear Transformation (**DLT**) algorithm





Intrinsic Calibration RMS=0.844679 pixel

The Gold Standard 3D DLT Algorithm

Objective

Given $n \geq 6$ world to image point correspondences $\{\mathbf{X}_i \leftrightarrow \mathbf{x}_i\}$, determine the Maximum Likelihood estimate of the camera projection matrix $\mathtt{P}$, i.e. the $\mathtt{P}$ which minimizes $\sum_i d(\mathbf{x}_i, \mathtt{P}\mathbf{X}_i)^2$.

Algorithm

(i) **Linear solution.** Compute an initial estimate of $\mathtt{P}$ using a linear method such as algorithm 4.2(p109):

(a) **Normalization:** Use a similarity transformation $\mathtt{T}$ to normalize the image points, and a second similarity transformation $\mathtt{U}$ to normalize the space points. Suppose the normalized image points are $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$, and the normalized space points are $\tilde{\mathbf{X}}_i = \mathtt{U}\mathbf{X}_i$.

(b) **DLT:** Form the $2n \times 12$ matrix $\mathtt{A}$ by stacking the equations (7.2) generated by each correspondence $\tilde{\mathbf{X}}_i \leftrightarrow \tilde{\mathbf{x}}_i$. Write $\mathbf{p}$ for the vector containing the entries of the matrix $\tilde{\mathtt{P}}$. A solution of $\mathtt{A}\mathbf{p} = \mathbf{0}$, subject to $\|\mathbf{p}\| = 1$, is obtained from the unit singular vector of $\mathtt{A}$ corresponding to the smallest singular value.

(ii) **Minimize geometric error.** Using the linear estimate as a starting point minimize the geometric error (7.4):

$$\sum_i d(\tilde{\mathbf{x}}_i, \tilde{\mathtt{P}}\tilde{\mathbf{X}}_i)^2$$

over $\tilde{\mathtt{P}}$, using an iterative algorithm such as Levenberg–Marquardt.

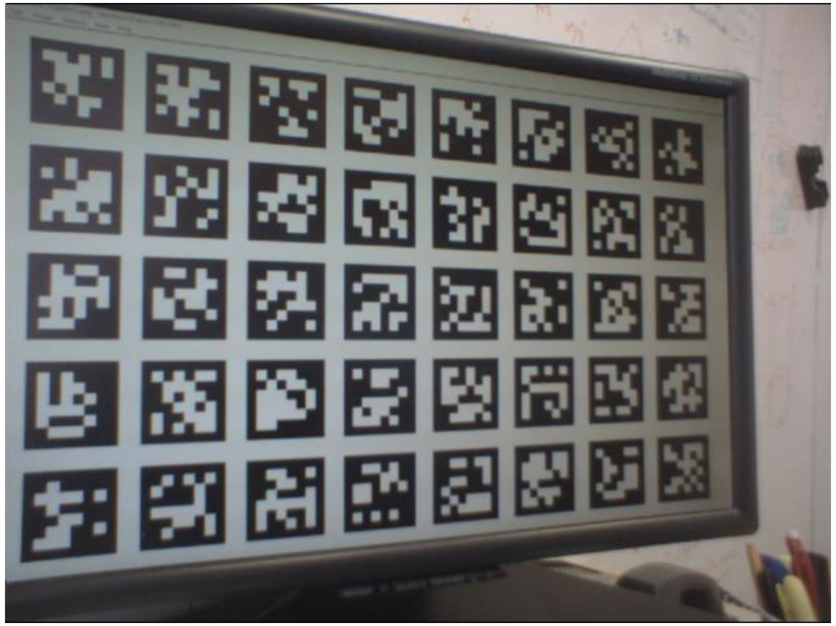
(iii) **Denormalization.** The camera matrix for the original (unnormalized) coordinates is obtained from $\tilde{\mathtt{P}}$ as

$$\mathtt{P} = \mathtt{T}^{-1}\tilde{\mathtt{P}}\mathtt{U}.$$

Hartley & Zisserman 2011

Calibration with a 2D Rig

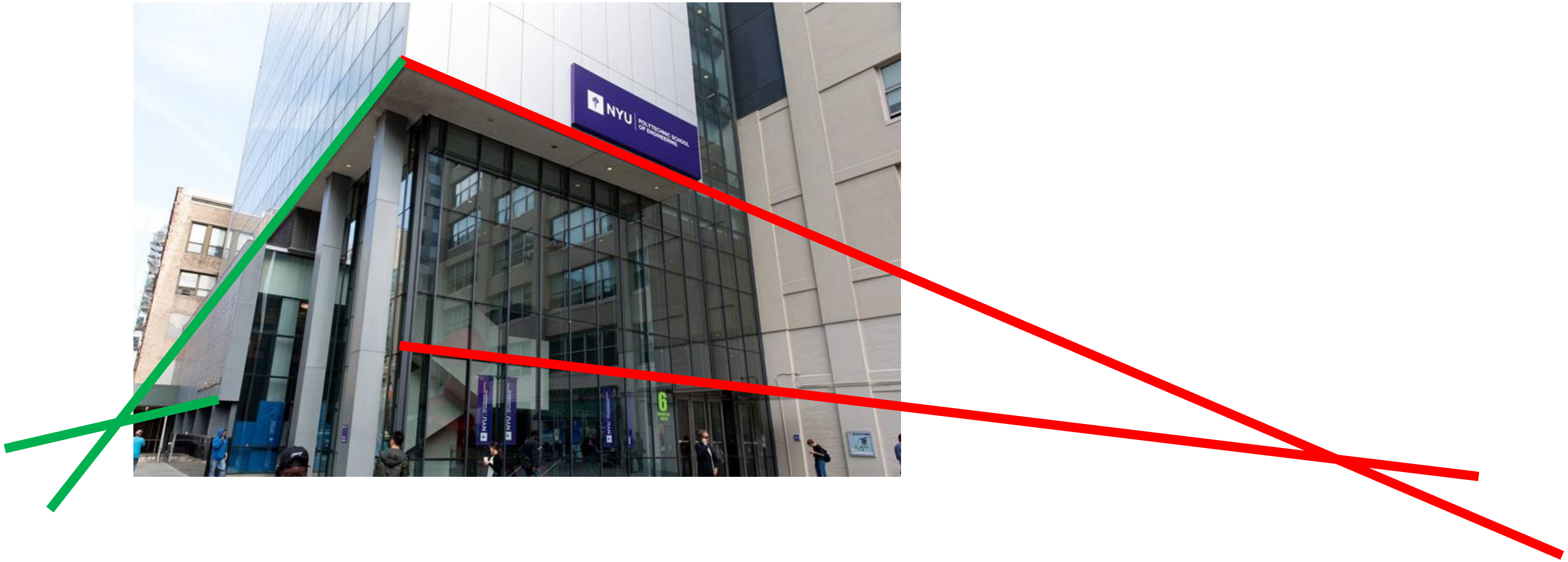
- Precisely measure each marker's 3D positions could be difficult
  - Usually need a total station
- An easier way is to use a 2D rig
  1. All markers on a same plane
  2. Measure each marker's 2D position
  3. Take multiple images
  4. Solve by Zhang's method
  5. Refine by bundle adjustment
- Advantages
  - Measuring 2D position is easy
  - Easy to setup planar rig



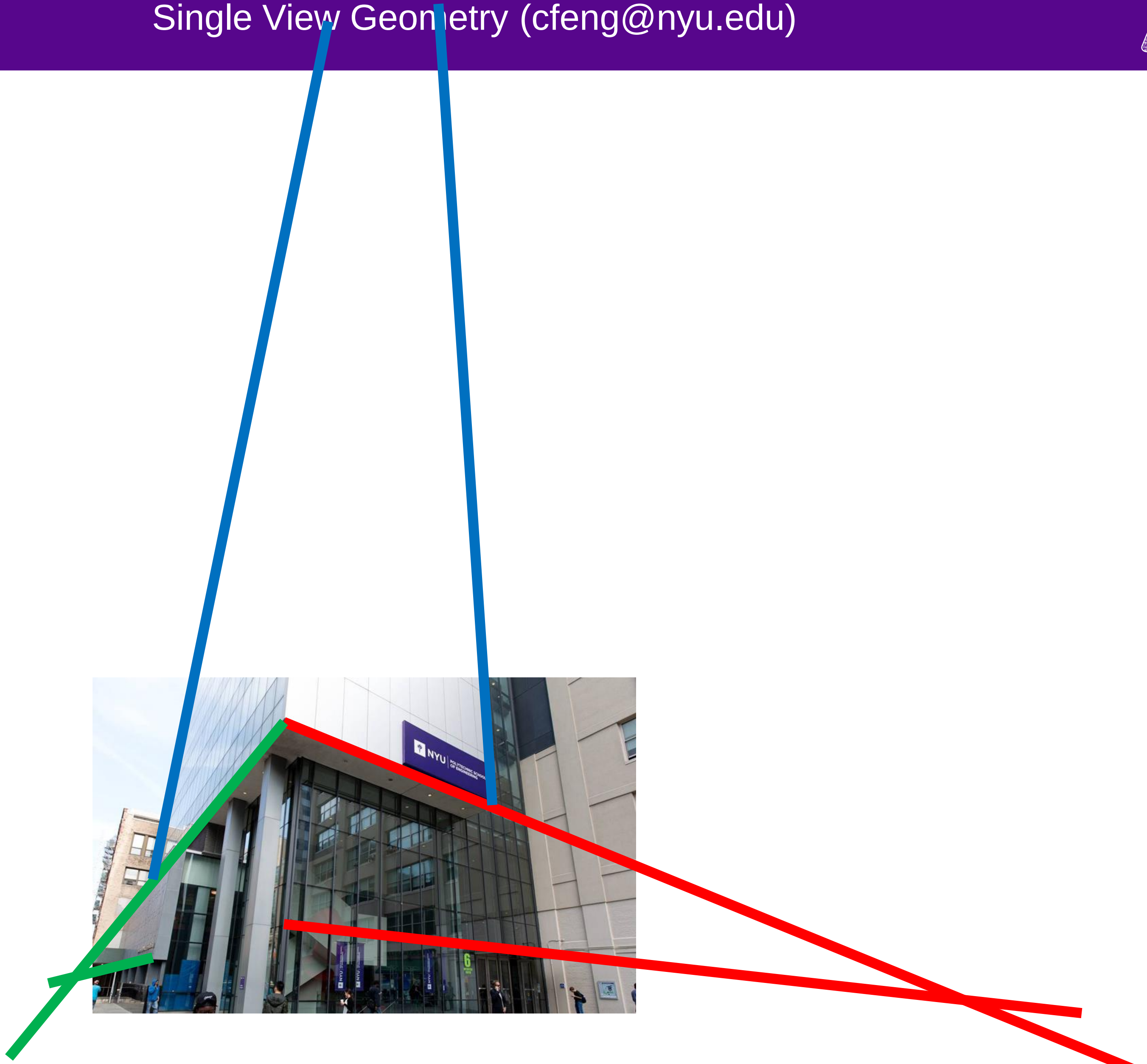
Z. Zhang, "A flexible new technique for camera calibration'", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.22, No.11, pages 1330–1334, 2000

Vanishing Point

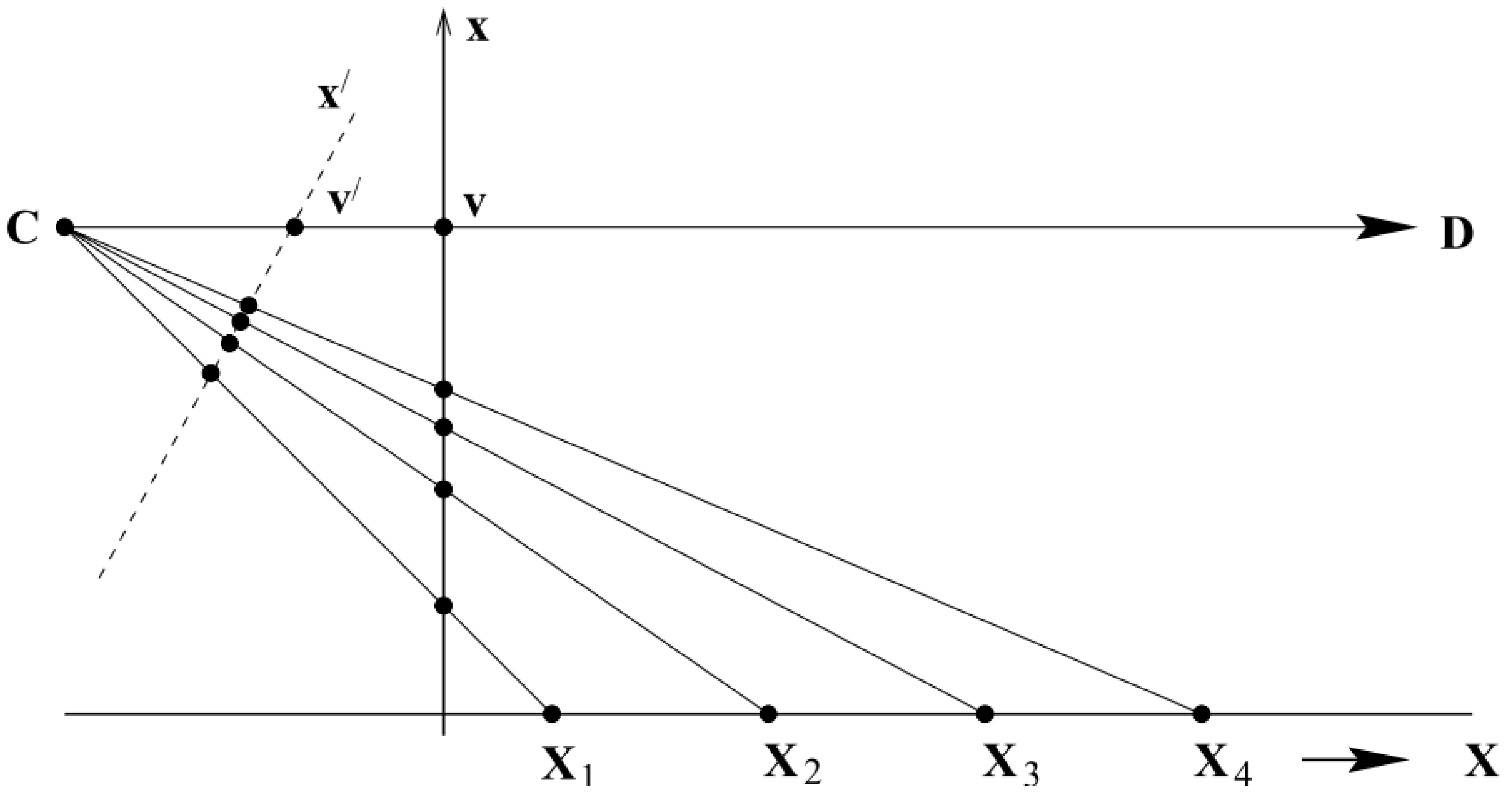
Vanishing Points



Vanishing Points



Vanishing Point – 1D



Hartley & Zisserman 2003

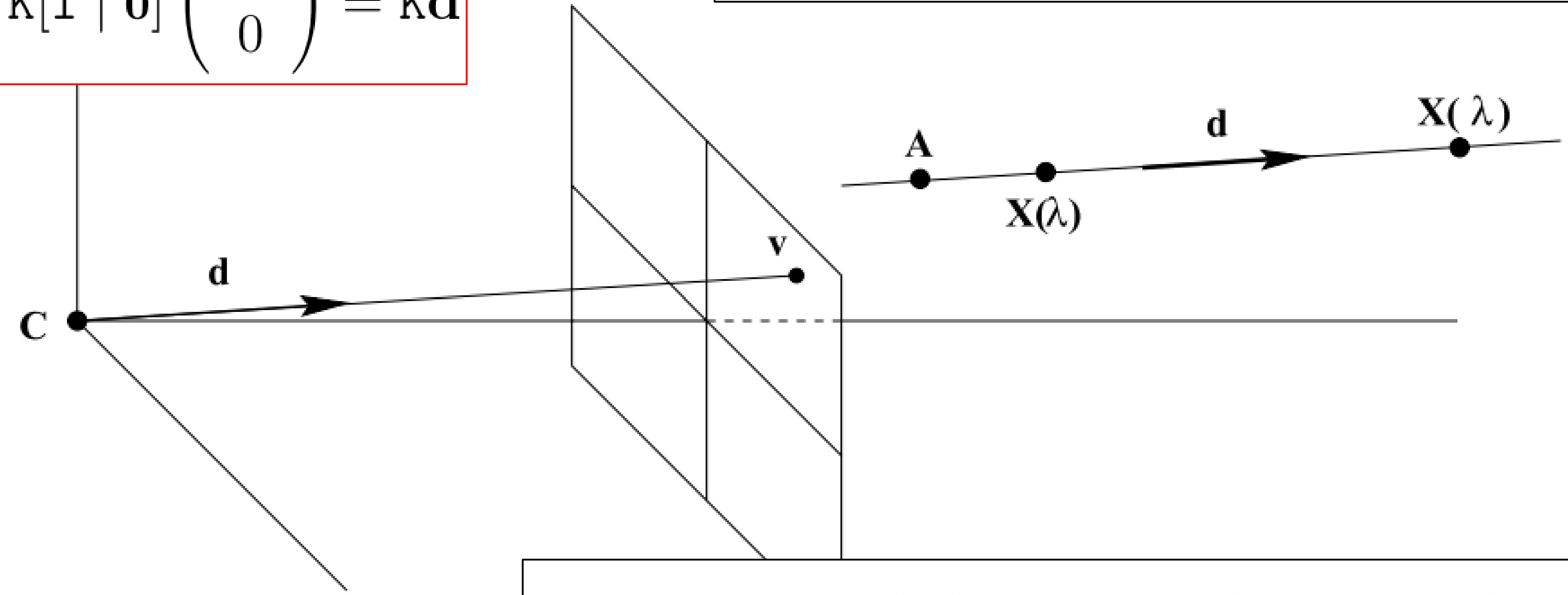
Vanishing Point – 3D

$$\mathbf{X}(\lambda) = \mathbf{A} + \lambda\mathbf{D}$$

$$\mathbf{D} = (\mathbf{d}^{\mathsf{T}}, 0)^{\mathsf{T}}$$

$$\mathbf{x}(\lambda) = \mathtt{P}\mathbf{X}(\lambda) = \mathtt{P}\mathbf{A} + \lambda\mathtt{P}\mathbf{D} = \mathbf{a} + \lambda\mathtt{K}\mathbf{d}$$

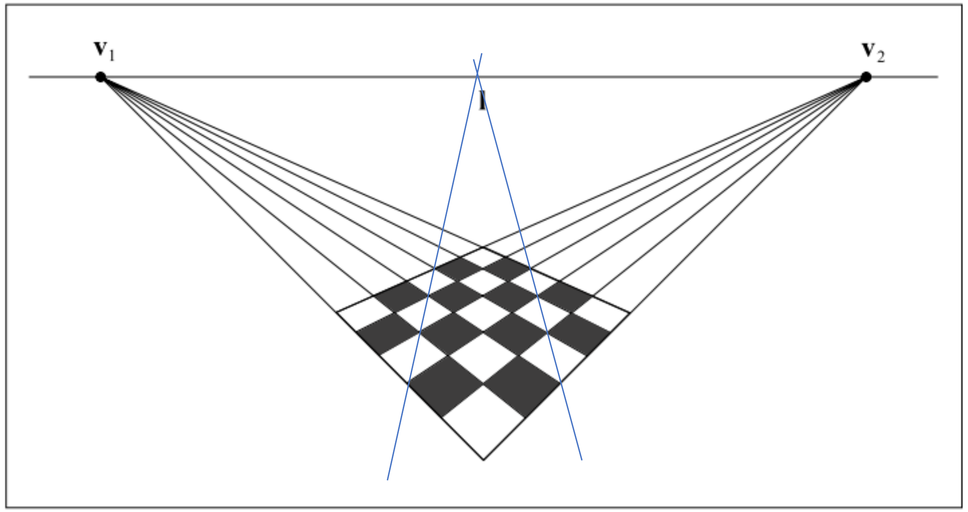
$$\mathbf{v} = \mathtt{P}\mathbf{X}_{\infty} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]\begin{pmatrix}\mathbf{d}\\0\end{pmatrix} = \mathtt{K}\mathbf{d}$$

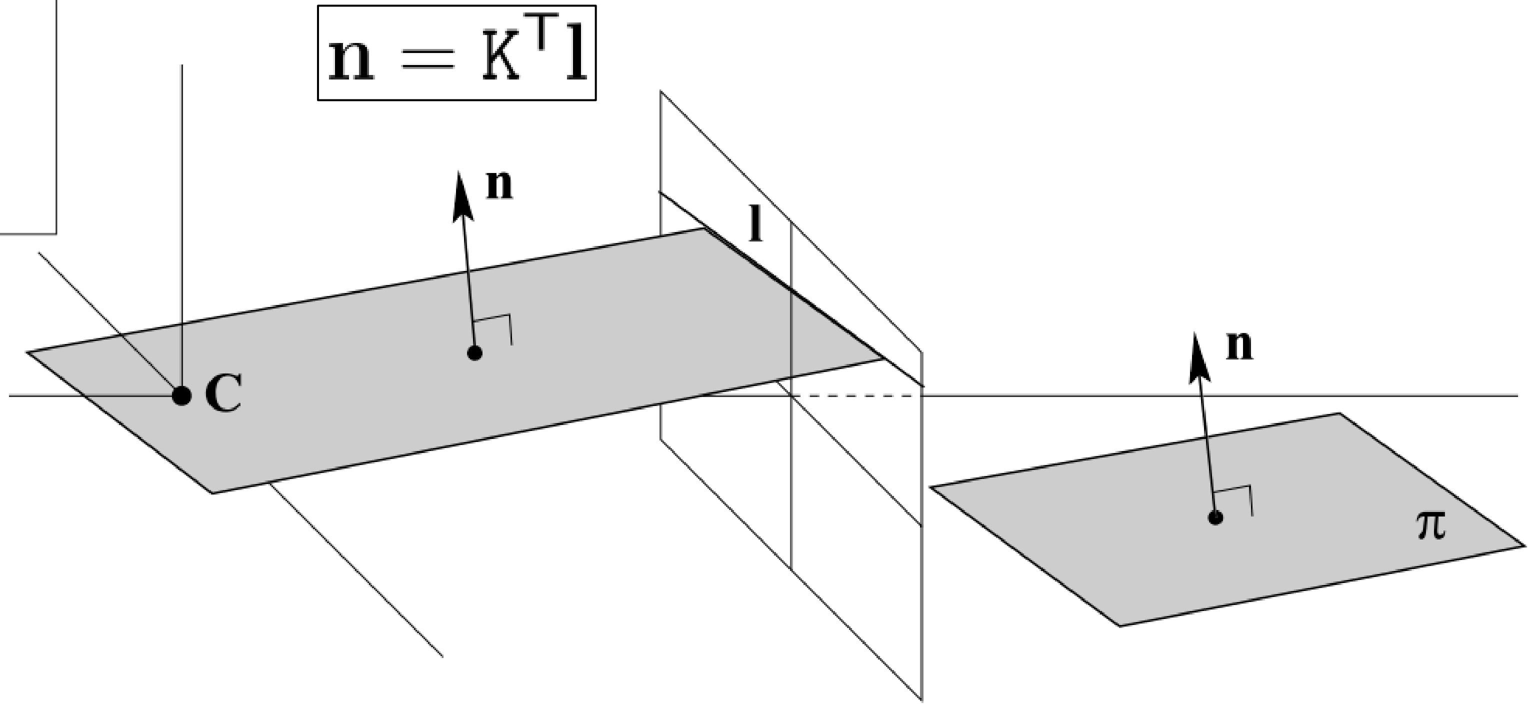


$$\mathbf{v} = \lim_{\lambda\to\infty} \mathbf{x}(\lambda) = \lim_{\lambda\to\infty} (\mathbf{a} + \lambda\mathtt{K}\mathbf{d}) = \mathtt{K}\mathbf{d}$$

Hartley & Zisserman 2003

Vanishing Line





Hartley & Zisserman 2003

Applications of Vanishing Points

- Rotation estimation of calibrated camera
  - vanishing point + calibration matrix == 3D direction

$$\mathbf{d}_i = \mathrm{K}^{-1}\mathbf{v}_i / \|\mathrm{K}^{-1}\mathbf{v}_i\|$$

$$\mathbf{d}_i' = \mathrm{R}\mathbf{d}_i$$

  - n (≥ 2) corresponding vanishing points

- **A calibrated camera is a protractor!**

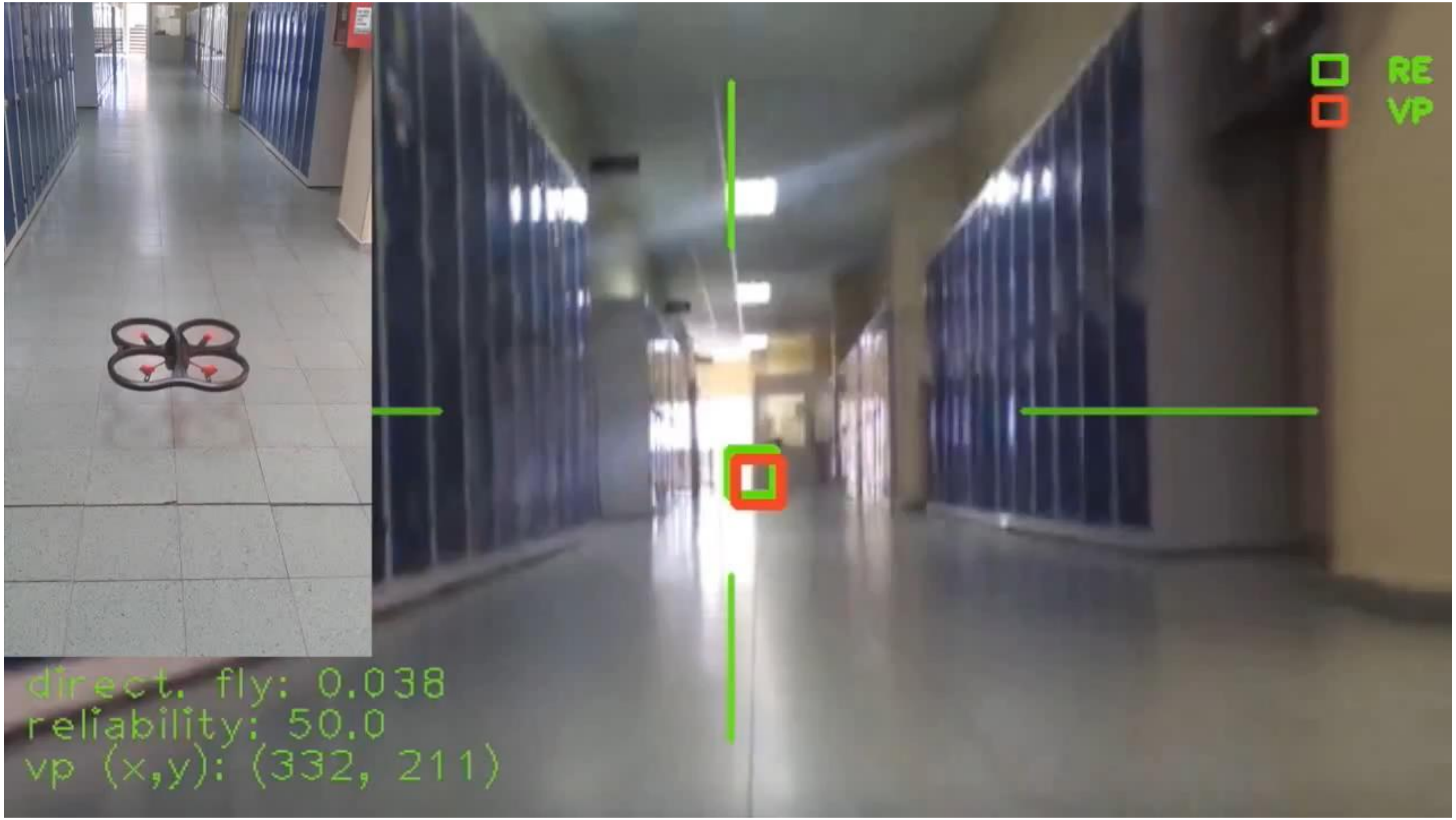
- Robot navigation/control
- Camera calibration/traffic surveillance

Hartley & Zisserman 2003

Vanishing Point for Robots

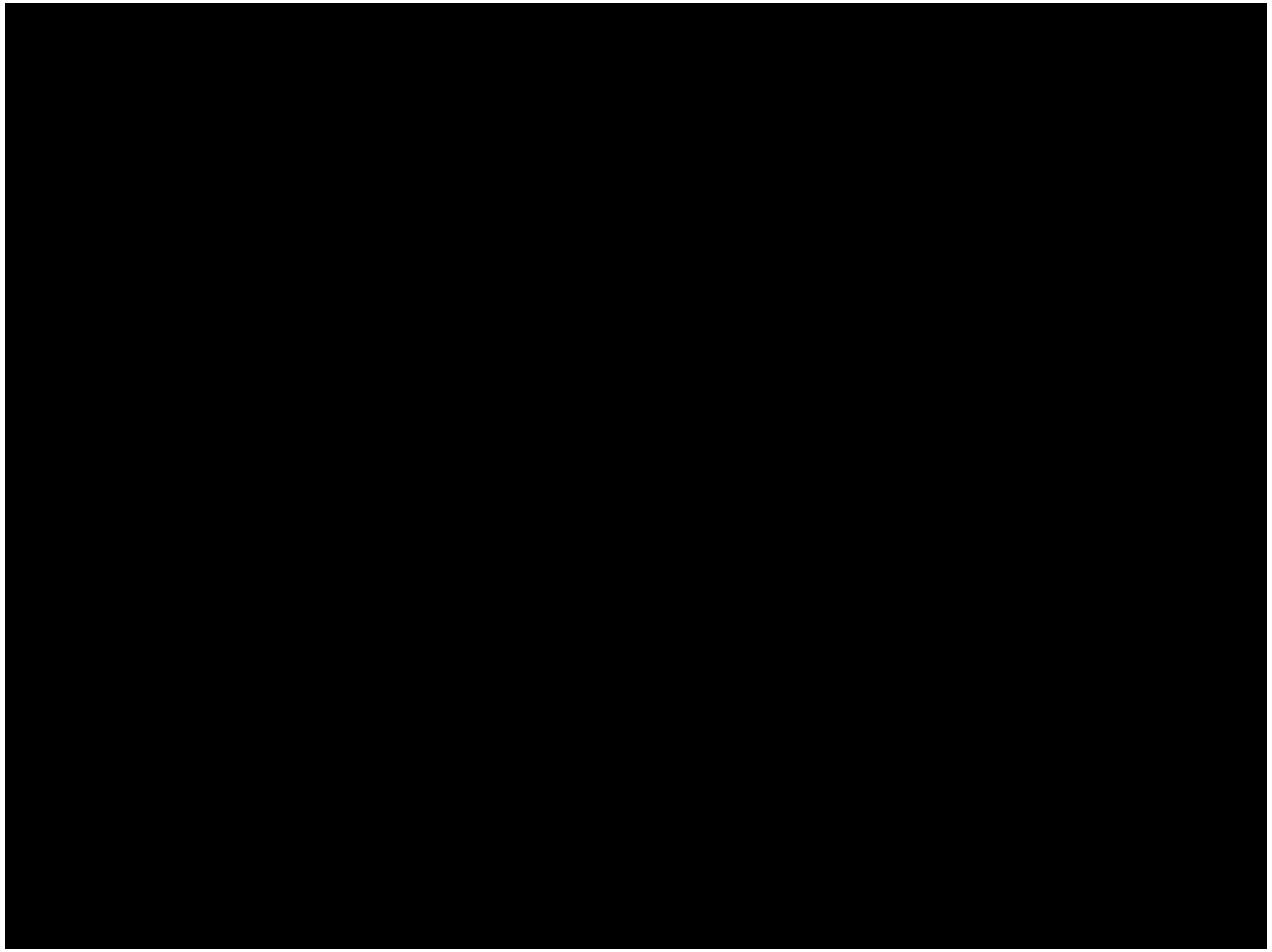
Video from: https://youtu.be/NC7mKUrJOE0

Vanishing Point for Robots



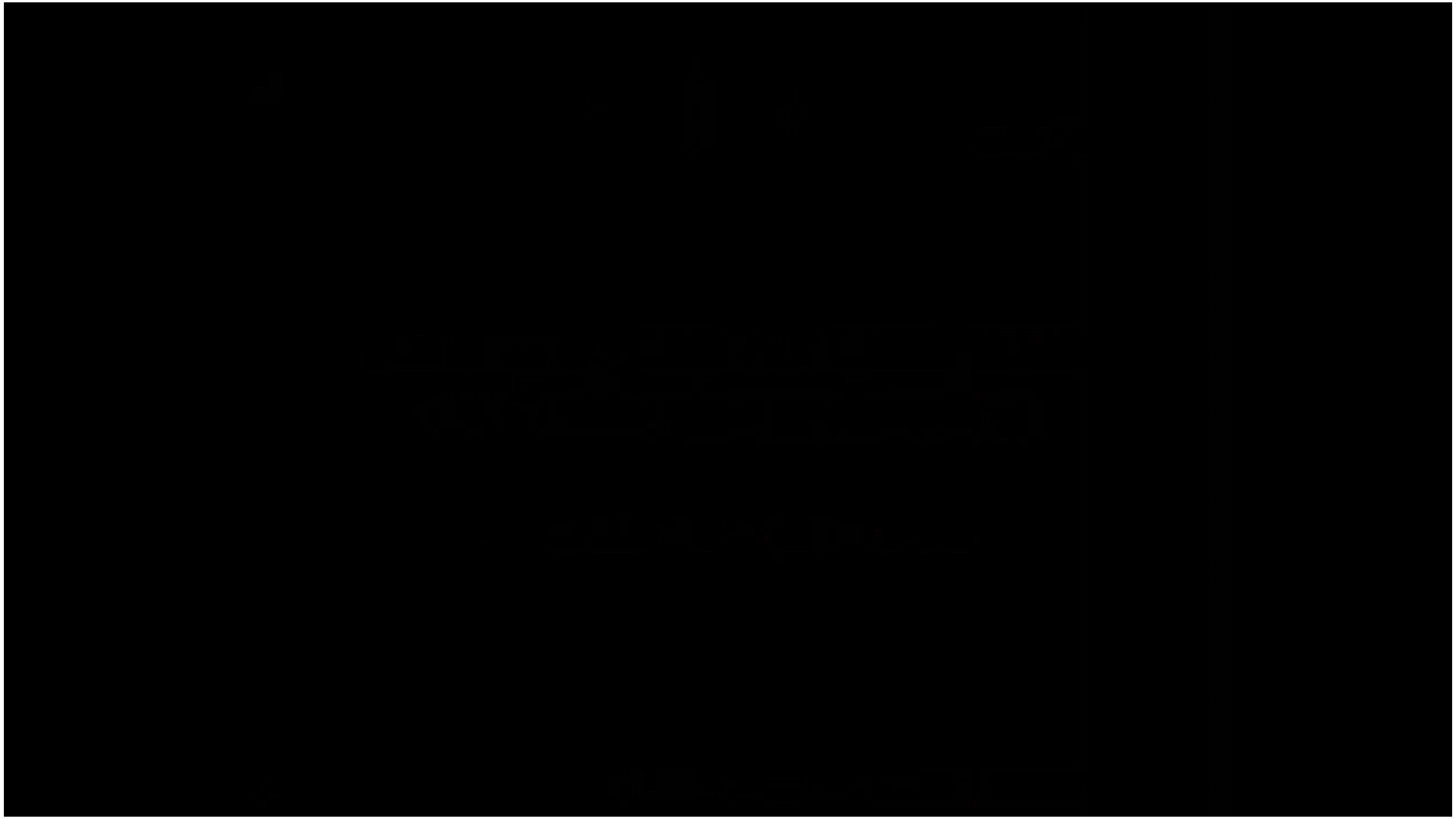
Video from: https://youtu.be/Te4AHOBHbiY

Vanishing Point for Autonomous Driving



Video from: https://youtu.be/Iws6-q9ji0o

Vanishing Point for Traffic Surveillance



Video from: https://youtu.be/S3msCdn3fNM

Simple Camera Calibration from 3 "Orthogonal" Vanishing Points

- Simplified camera

$$K = \begin{bmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Three vanishing points
  - Mutually orthogonal in 3D

$$\begin{bmatrix} \lambda_1 u_1 & \lambda_2 u_2 & \lambda_3 u_3 \\ \lambda_1 v_1 & \lambda_2 v_2 & \lambda_3 v_3 \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} = \mathbf{P} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{R}$$

- Using orthonormal constraints

$$\mathbf{R} = \begin{bmatrix} \lambda_1(u_1 - x_0)/f & \lambda_2(u_2 - x_0)/f & \lambda_3(u_3 - x_0)/f \\ \lambda_1(v_1 - y_0)/f & \lambda_2(v_2 - y_0)/f & \lambda_3(v_3 - y_0)/f \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix}$$

=> $\lambda_1\lambda_2((u_1 - x_0)(u_2 - x_0)/f^2 + (v_1 - y_0)(v_2 - y_0)/f^2 + 1) = 0$

- Get rid of non-zero unknown scale factors

$$(u_2 - u_3)(u_1 - x_0) + (v_2 - v_3)(v_1 - y_0) = 0,$$
$$(u_1 - u_2)(u_3 - x_0) + (v_1 - v_2)(v_3 - y_0) = 0.$$

Simple Camera Calibration from 3 “Orthogonal” Vanishing Points

- Solve a 2x2 equation

$$
\begin{aligned}
\mathbf{A}\mathbf{x} &= \mathbf{b}, \\
\mathbf{A} &= \begin{bmatrix} u_1 - u_3 & v_1 - v_3 \\ u_2 - u_3 & v_2 - v_3 \end{bmatrix}, \\
\mathbf{x} &= \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}, \\
\mathbf{b} &= \begin{bmatrix} (u_1 - u_3)u_2 + (v_1 - v_3)v_2 \\ (u_2 - u_3)u_1 + (v_2 - v_3)v_1 \end{bmatrix}
\end{aligned}
$$

- Compute focal length

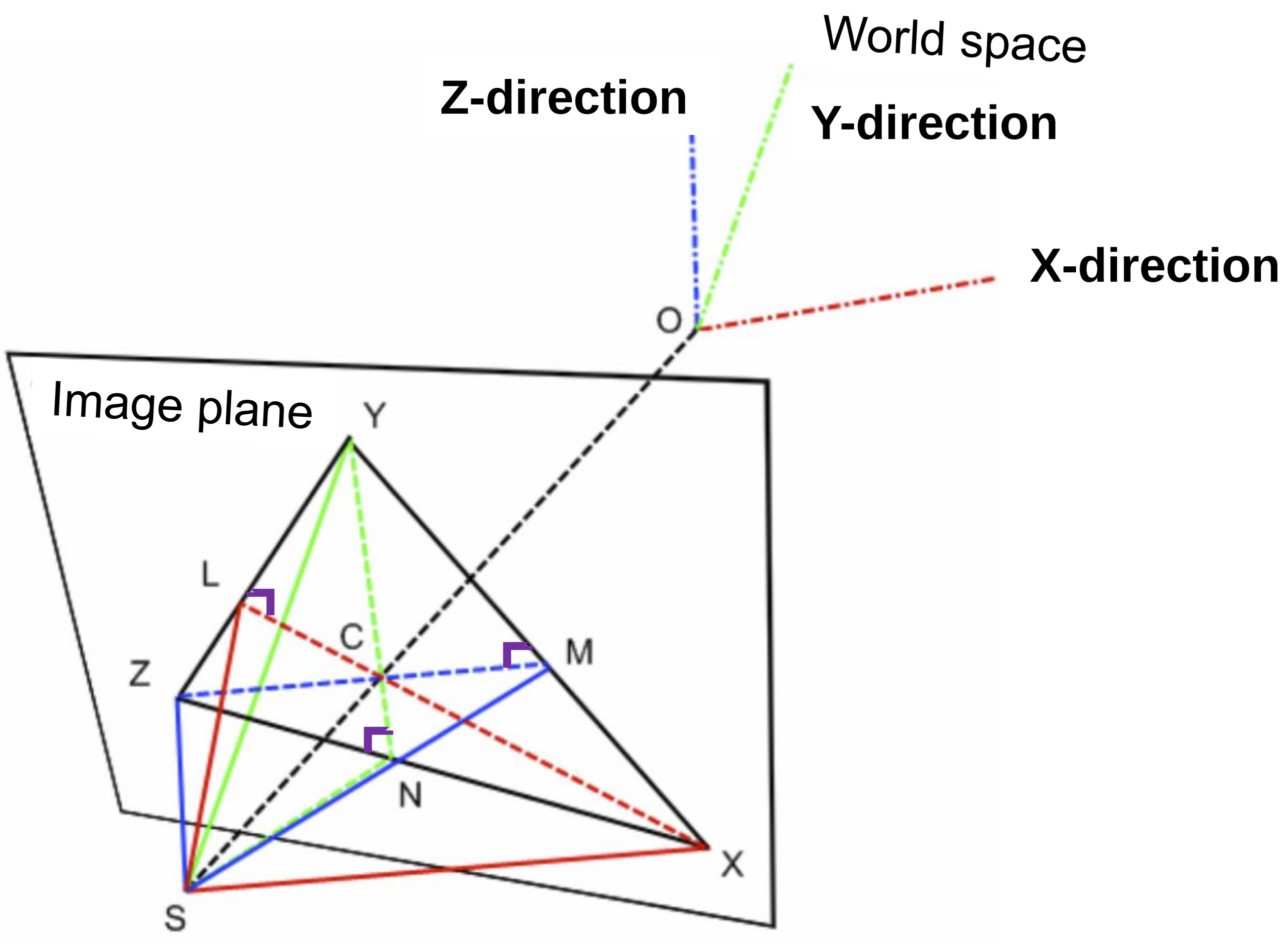
$$f = \sqrt{-(u_1 - x_0)(u_2 - x_0) - (v_1 - y_0)(v_2 - y_0)}.$$

Vanishing Point Calibration: Geometric Explanation

- X/Y/Z: three VPs on the image
- S: projection center
- C: principal point
  - SC: optical axis

- **C is the orthocenter of the triangle XYZ**

- =>SXYZ: Trirectangular Tetrahedron
- => XY⊥ ZM
- => C is ortho-center of XYZ



Next Week

* Hands-on: AprilTag & camera calibration
+ Epipolar geometry
++ Fundamental matrix
+ Essential matrix
+ Planar Homography
+ PnP problem
++ Hand-eye calibration

*: know how to code
++: know how to derive
+: know the concept

References for Next Week

- HZ2003:
  - Section 9.1, 9.2, 9.3, 9.5, 9.6, 11.1, 11.2, 11.7
- Co2017:
  - Section 14.2, 11.2.3
- Sz2022:
  - Section 11.3, 11.2, 12.1
- FP2011:
  - Section 7.1, 8.1.2
- Radu Horaud, Fadi Dornaika. Hand-eye Calibration. International Journal of Robotics Research, SAGE Publications, 1995, 14 (3), pp.195–210.