Univerza v Ljubljani
Fakulteta za elektrotehniko

# Computer Vision 13 – Multiple view geometry, 3D and stereo

doc. dr. Janez Perš
(with contributions by prof. Stanislav Kovačič)

Laboratory for Machine Intelligence
Faculty of Electrical Engineering
University of Ljubljana

# Regarding the exam

- Mandatory:
  - Check that you can log into **https://studij.fe.uni-lj.si**
  - Apply for an exam **at least five days** before exam date!
- 1st written exam date: Monday, 21.1. 18.00
- 2nd written exam date: Friday, 1.2. (time TBA)
- Results announced 2-3 days later
- Oral exam 1-2 days after results
- You **must** have your labs done and graded before sitting written exam!

# Q+A regarding the exam

Q: Will be there questions regarding Matlab/code?

A: No. Matlab is only a  tool we use for labs.

Q: Will there be theoretical questions in written exam?

A: Not in the sense “explain this in X sentences”. But not all questions will be purely computational. I may provide you with 2 graphs and ask you to draw the 3rd one, based on your knowledge of theory.

Q: What tools will be allowed during a written exam?

A: **Calculators** will be allowed, but NO PHONES.

Q: Which lectures should we study for the exam?

A: All of them. And the labs to refresh your knowledge.

# Quick recap of the previous lectures

- Image formation
- Image analysis
- Image alignment and matching

# Outline

- 3D sensing & perception
- Stereo vision
- Lateral stereo vision model
- Epipolar geometry
- Stereo matching

# Motivation

- With projection from 3D world to 2D image, one dimension (depth, range, object distance) is lost.
- Fundamental questions of CV:
  - How to recover 3D representation of scene, based on one, two, or more images
- Note:
  - Many CV problems can be solved without explicit 3D reconstruction!
  - (remember – until now, we did not talk about it!)

# Motivation

- Recovering 3D information is the domain of 3D computer vision, such as
  - Stereo Vision,
  - Photometric stereo,
  - Shape from X,
  - Structured Lighting, sometimes extended to “Active Stereo”
  - and “Structure from Motion” – SfM.

# 3D structure from 2D data

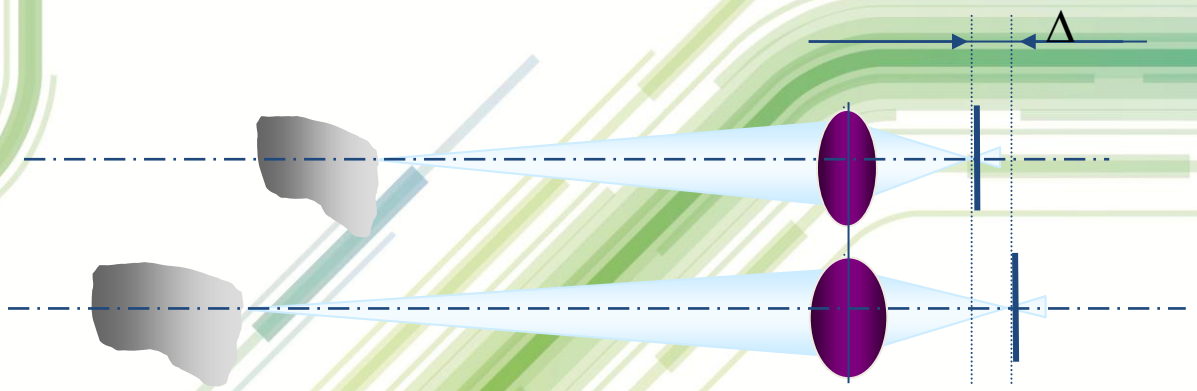
- 3D based on one image:
  - Focusing (Depth from Focus),
  - Zooming (Depth from Zooming),
  - Texture analysis (Depth from texture),
  - Shading (Depth from Shading), "Shape from X".
  - Time of Flight cameras

# 3D structure from 2D data

- 3D based on one image plus “controlled structured light”
  - Photometric stereo,
  - Range finders, or scanners.
- 3D from two or more images (“multiple view”):
  - Stereo vision, “stereo”
  - Active stereo (stereo + structured lighting)
  - Depth from Motion, and Structure from Motion - SfM.

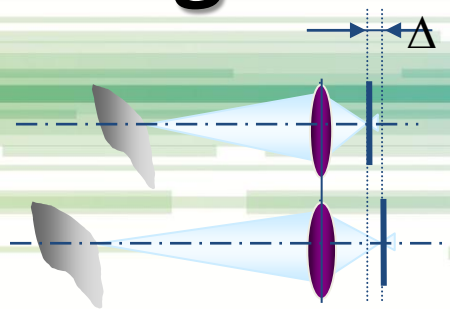
# 3D from focusing, zoom and texture

# 3D from one image - focusing



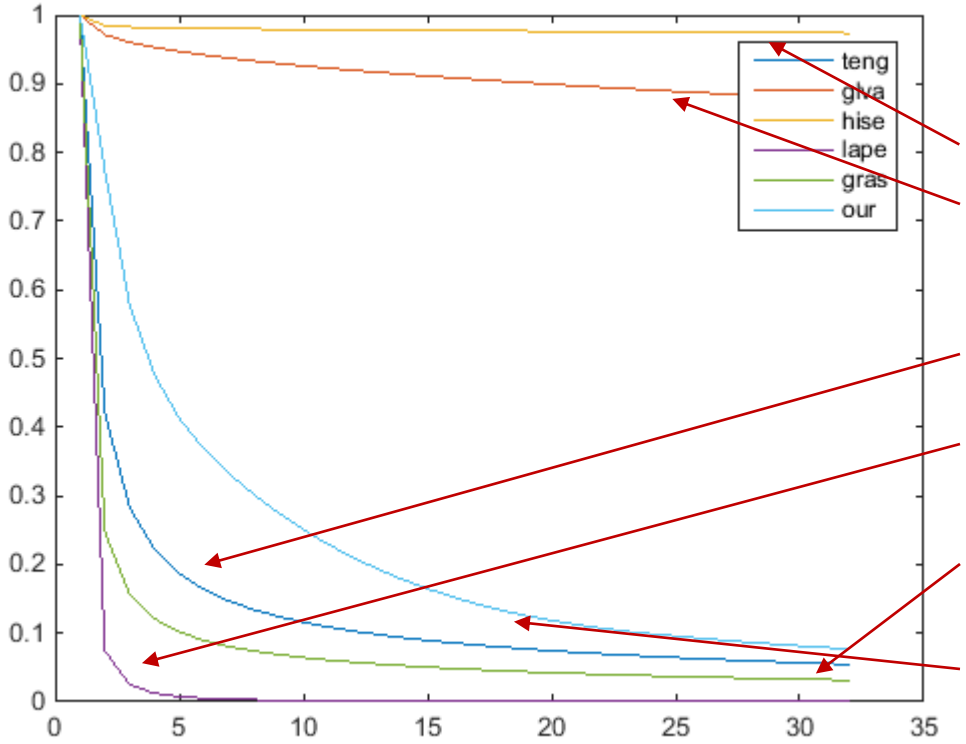

- When image is not in focus, it looks blurred, of low contrast, i.e., it *lacks high-frequency components*.
- That means that sensor is not in the imaging plane. If we could evaluate "out of focus", be able to bring the image ‚in focus', then, having calibrated lens, we should be able to recover object distance from Δ
- For depth recovery we need
  - Autofocusing, which needs
    - contrast measure
    - controlled lens/sensor positioning
  - calibrated lens

# 3D from one image - focusing




- Measures of focus:
  - grey value variance
    - if variance is high, diversity of grey values should be large, thus, “image is in focus”
  - entropy-based measures
    - compute grey value histogram, compute entropy
    - if entropy is high, contrast should be high, thus, “image is in focus”
  - gradient-based measures
    - compute gradient magnitude, sum over region
    - if gradient is high, image must be sharp, therefore, “image is in focus”
- Our work:
  - Prof. Matej Kristan (now at FRI) has devised well accepted focusing method while part of our group.
  - Check: http://vision.fe.uni-lj.si/RESEARCH/focus/index.html

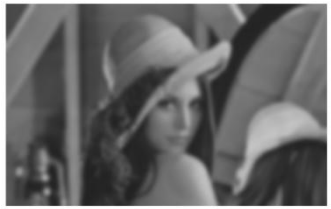
# Focusing – an example




Only a few, among many focus measures are shown!

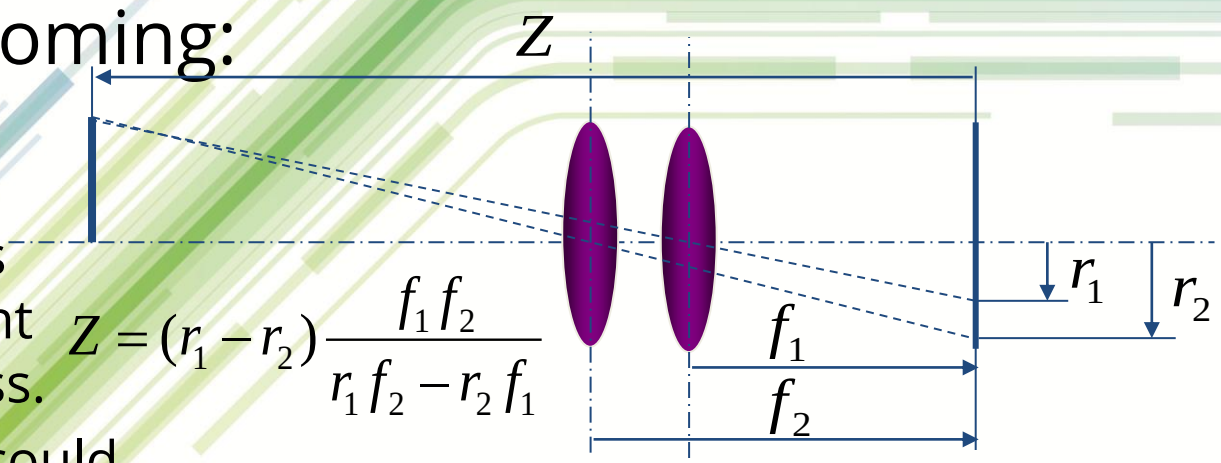
- **hise:** graylevel entropy
- **glva**: graylevel variance
- **teng**: (mean) sum of grad magnitude squared (sobel)
- **lape**: laplacian energy, mean of laplacian squared
- **gras**: (mean) sum of squared x derivatives
- **our**: Bayes entropy and DCT
- Focus measures based on derivatives are simple and work well.

Progressively blurred images



# 3D from one image - zooming

- Depth from zooming:
  - due to zooming pixels in image ‚move', but pixels from more distant objects move less.
  - Therefore, if we could estimate these displacements in terms of depth, we should be able to recover depth.

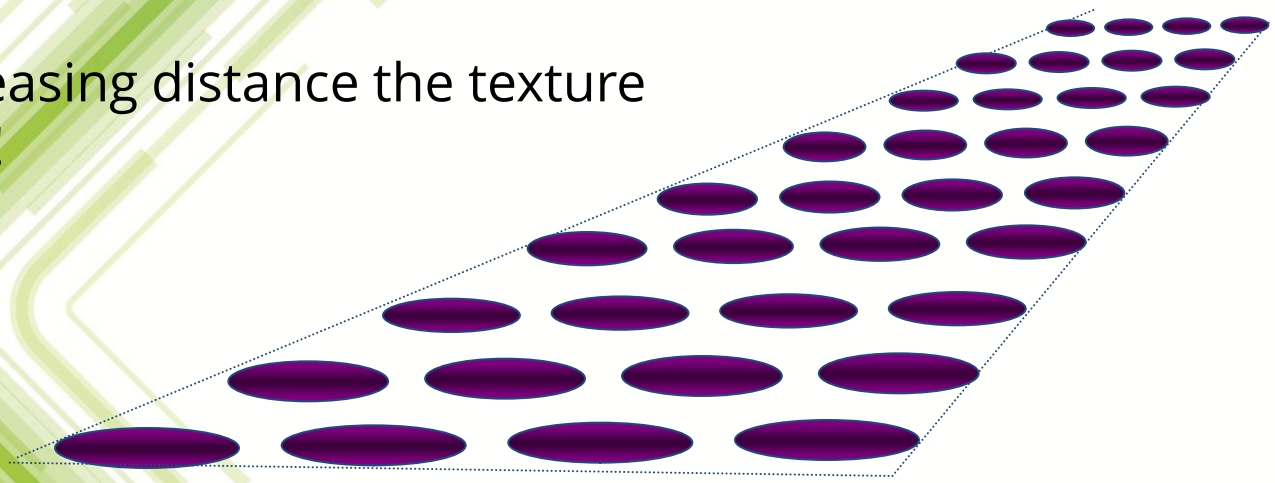



$$Z = (r_1 - r_2)\frac{f_1 f_2}{r_1 f_2 - r_2 f_1}$$

- What is needed:
  - to evaluate relative movement of points, correspondence problem (e.g. optical flow, matching), should be solved, motorized and calibrated
  - zoom lenses are needed.

# 3D from focus, zoom and texture

- Note:
  - depth from focus and depth from zoom are more of qualitative value, rather that quantitative, e.g., this object is closer, or more distant, ….
- Another qualitative technique is “depth from texture”
  - With increasing distance the texture gets finer!



# 3D from structured lighting

# 3D from structured lighting

- Terminology:
  - (1) Structured lighting, Laser range finding, LIDAR (Light Detection and Ranging)
  - (2) Active stereo - just like two-camera stereo, but the stereo correspondence is simplified by the *controlled light source.*
  - All in all, it largely simplifies the *correspondence problem*

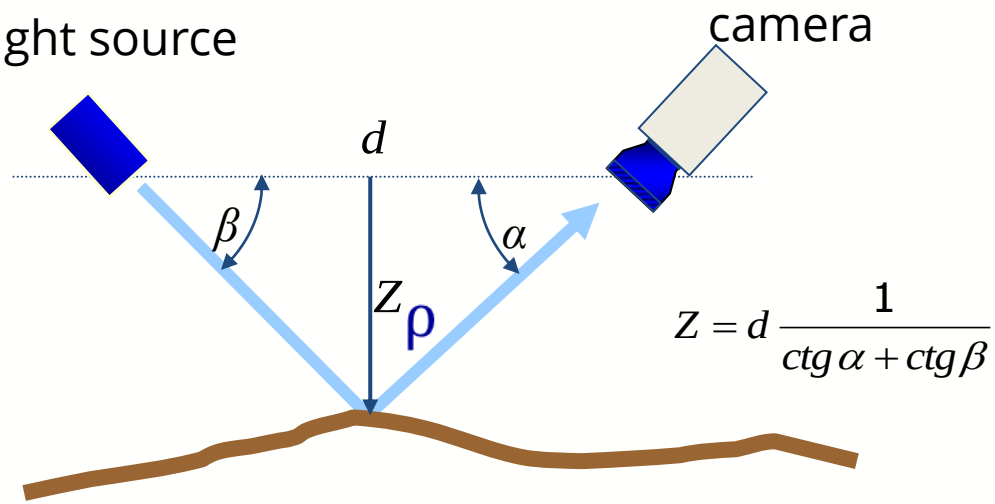
Polar coordinates: $(\rho, \beta)$

$Z = \rho \sin(\alpha)$

$X = \rho \cos(\alpha)$

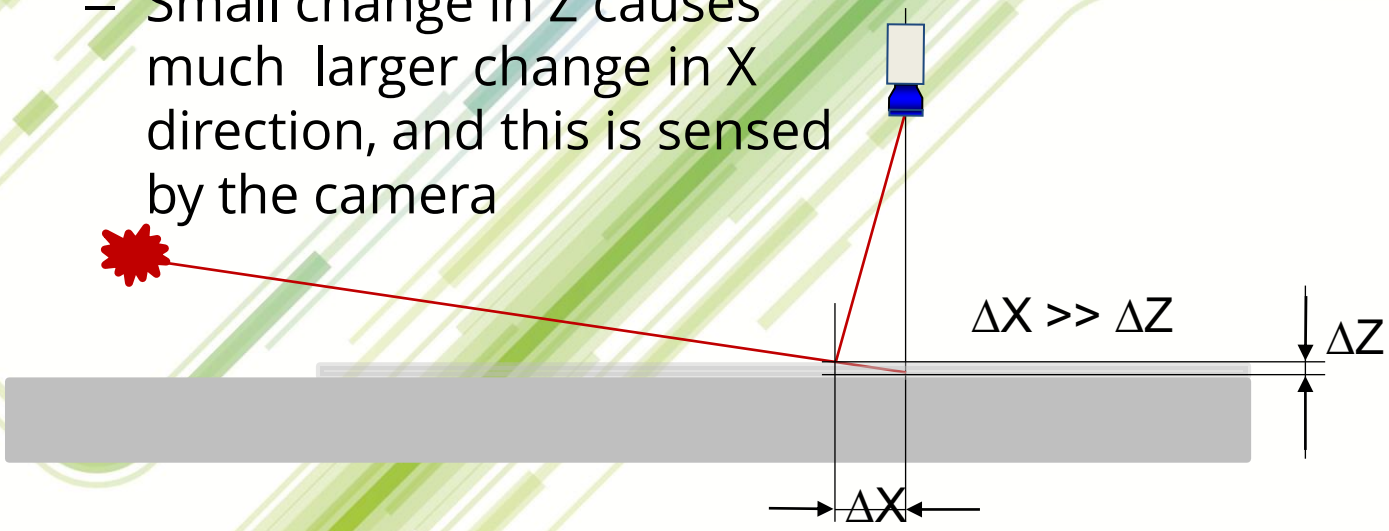
$\rho = d \sin(\beta) / \sin(\alpha+\beta)$

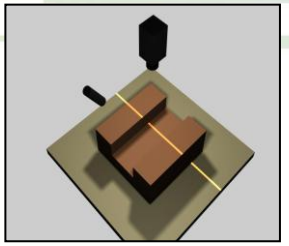
$Z = d \frac{\sin\alpha \sin\beta}{\sin(\alpha+\beta)}$

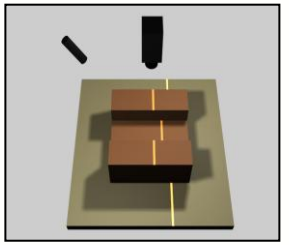


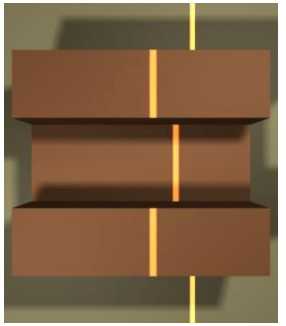
# 3D from structured lighting

- "Optical amplification"
  - Small change in Z causes much larger change in X direction, and this is sensed by the camera



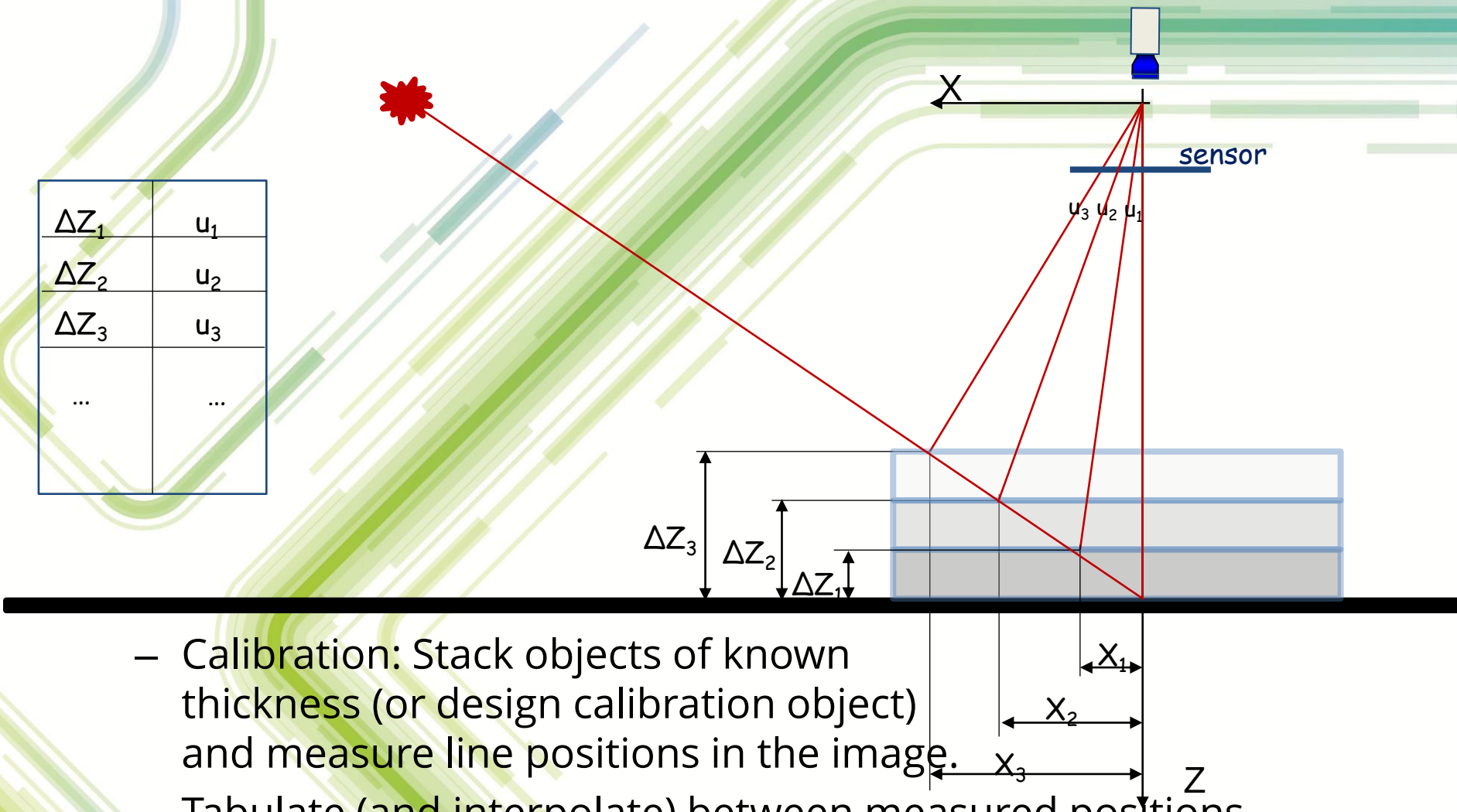




  - Extensively used in industry due to high measuring speed and accuracy, ~10 μm and better
  - In many industrial applications we only need to estimate the relative distances in depth
  - Therefore, the system can be calibrated in a simple way

# 3D from structured lighting

| $\Delta Z_1$ | $u_1$ |
|---|---|
| $\Delta Z_2$ | $u_2$ |
| $\Delta Z_3$ | $u_3$ |
| ... | ... |



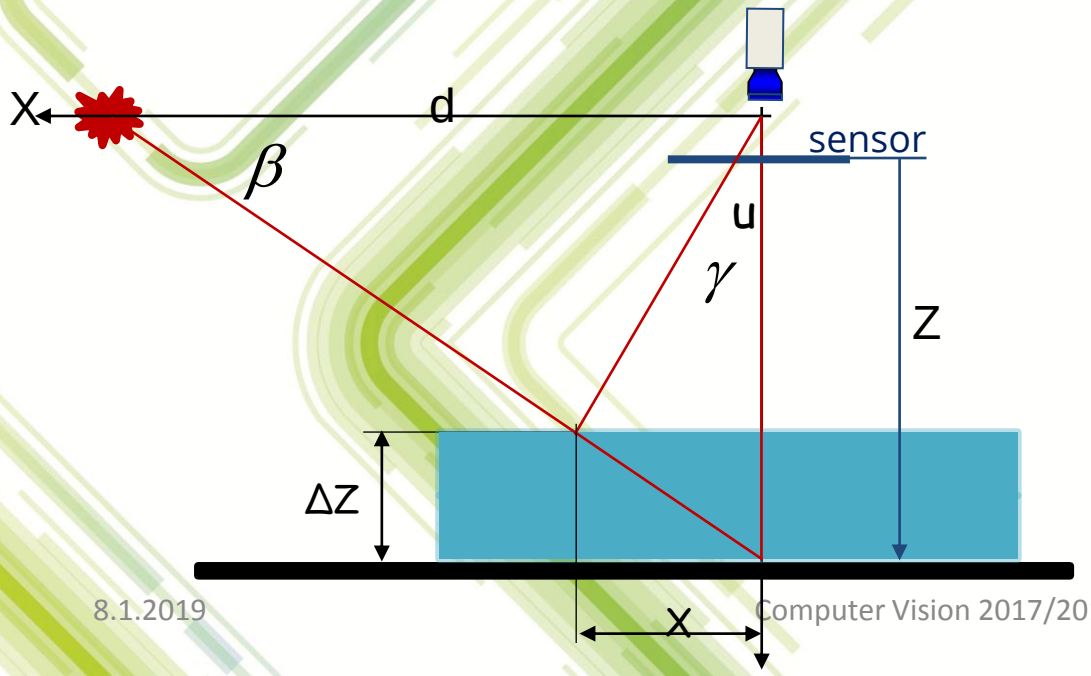

- Calibration: Stack objects of known thickness (or design calibration object) and measure line positions in the image.
- Tabulate (and interpolate) between measured positions

# 3D from structured lighting

$$\frac{X}{Z-\Delta Z}=\frac{u}{f}=tg\gamma \rightarrow X=(Z-\Delta Z)tg\gamma$$

$$Z=d\,tg\beta$$

$$\frac{\Delta Z}{X}=tg\beta \rightarrow \Delta Z=X\,tg\beta=(Z-\Delta Z)tg\gamma tg\beta \rightarrow \Delta Z=\frac{Z\,tg\gamma tg\beta}{1+tg\gamma tg\beta}=\frac{d\,tg\gamma}{1+tg\gamma tg\beta}$$




To measure depth at the exact position, exchange the position of camera and light source (equivalent setup!)

# Structured lighting



SICK

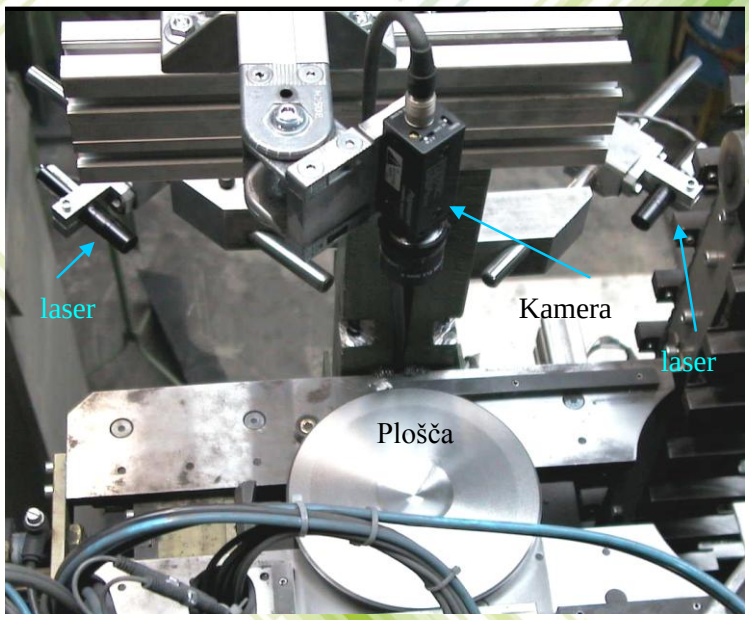
KEYENCE

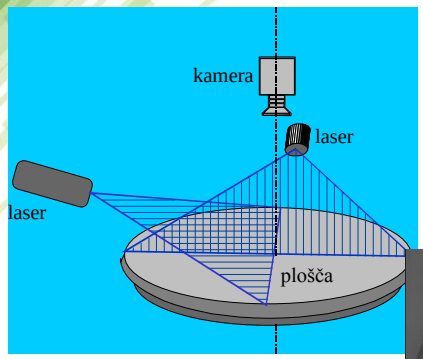
Various solutions on the market!

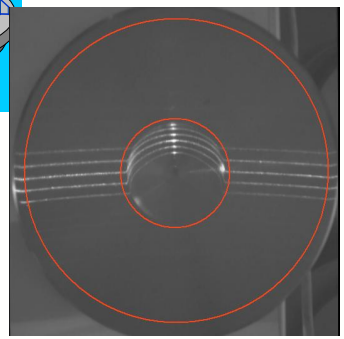
SmartRay

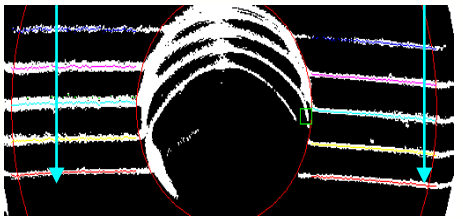
# Examples of custom design

- Concavity grading 0° - 0,5° in steps of 0,05°



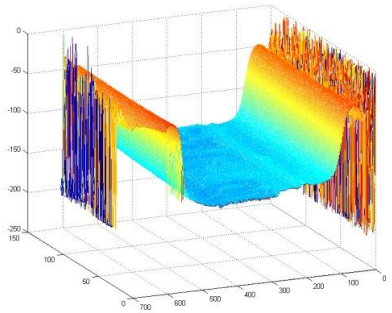


Images by Franci Lahajnar

# Examples of custom design

- Measuring profiles





# Space-coded structured light

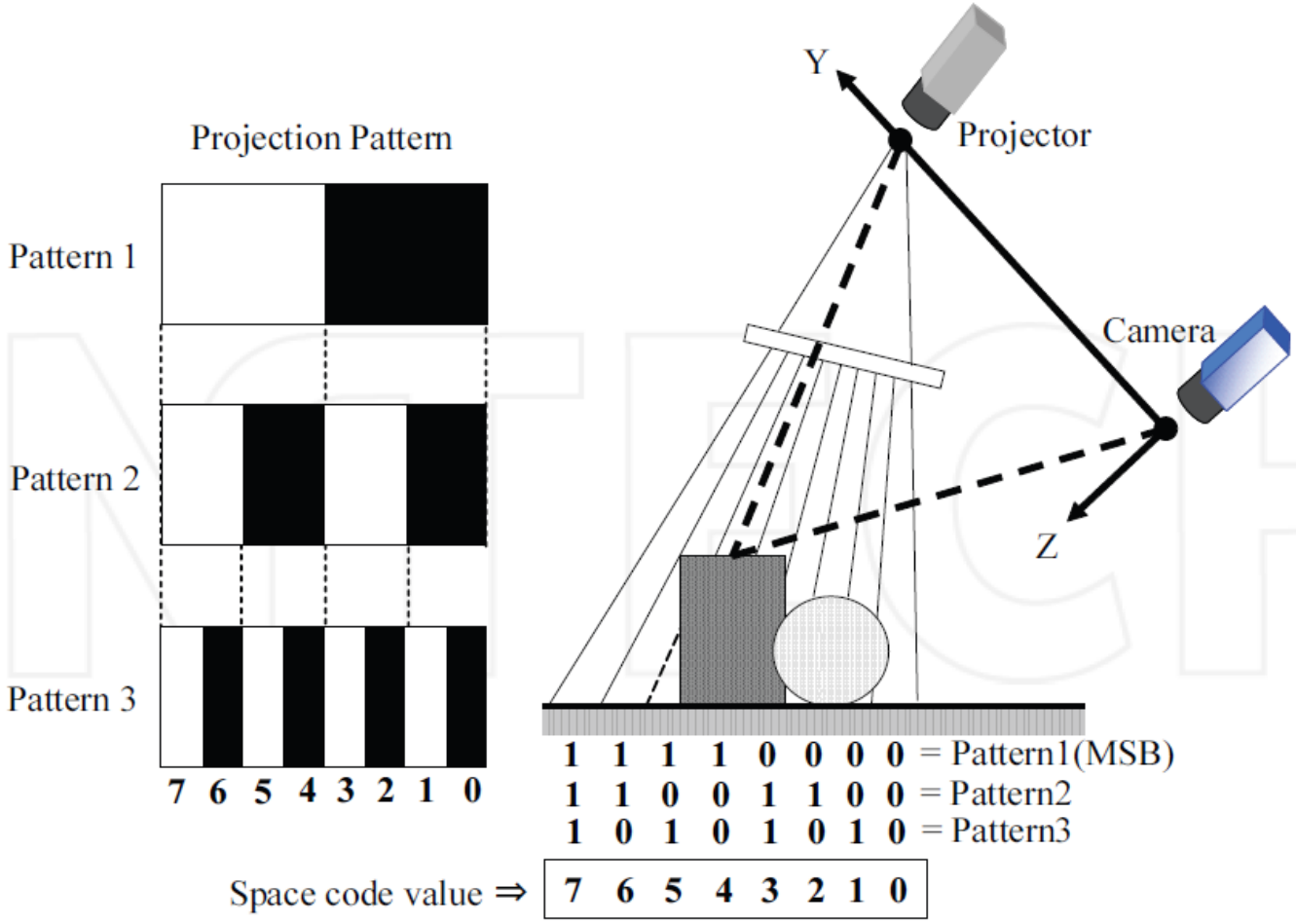
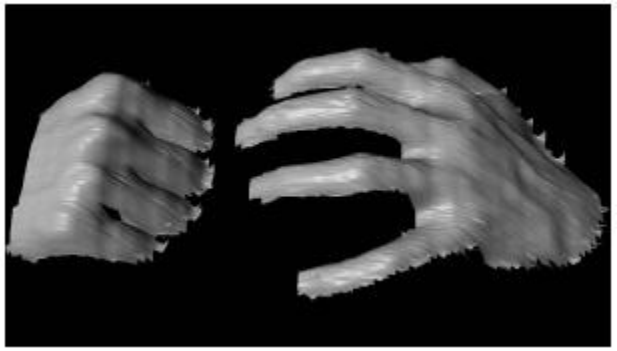



Fig. 1. Coded structured light projection method.

Source: http://cdn.intechopen.com/pdfs-wm/32185.pdf

# Color-coded structured light





Color coding and space coding simplify/alleviate the correspondence problem.

L. Zhang, B. Curless, and S. M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. *3DPVT* 2002

# Kinect 360: Structured infrared light


Source: http://bbzippo.wordpress.com/2010/11/28/kinect-in-infrared

- Note that this is the “old” Kinect
- “new” Kinect (Kinect v2, Kinect for Windows) uses *entirely different* technology from the different company

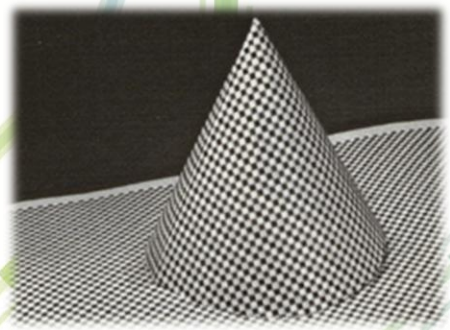
# Kinect V2: Time of flight camera

- Also known as “Kinect for windows”, Kinect for Xbox One.


  - It emits modulated infrared laser light
  - It is able to measure *how long does it take for light to travel to the object and back.*
  - It is able to measure this for *each of the pixels!*
  - The time delay (“time of flight”) directly translates into the distance of object from the sensor for each pixel!

# Stereo vision

# Task of stereo vision

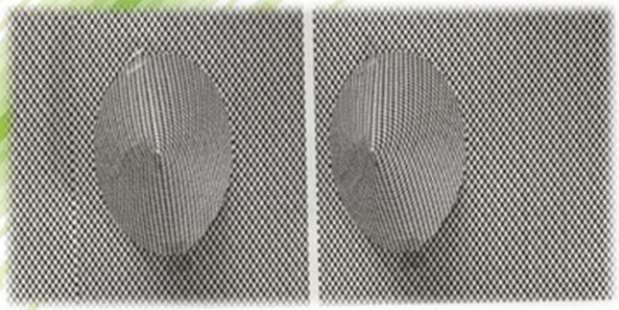


**I.**

This is the scene, with
3D object of a particular interest

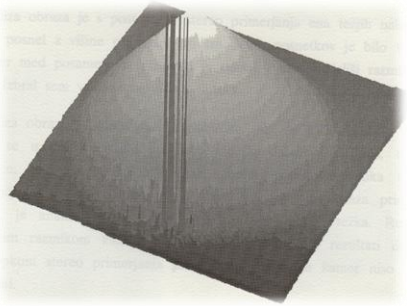
**II.**

Take two (or more)
2D images of
that scene

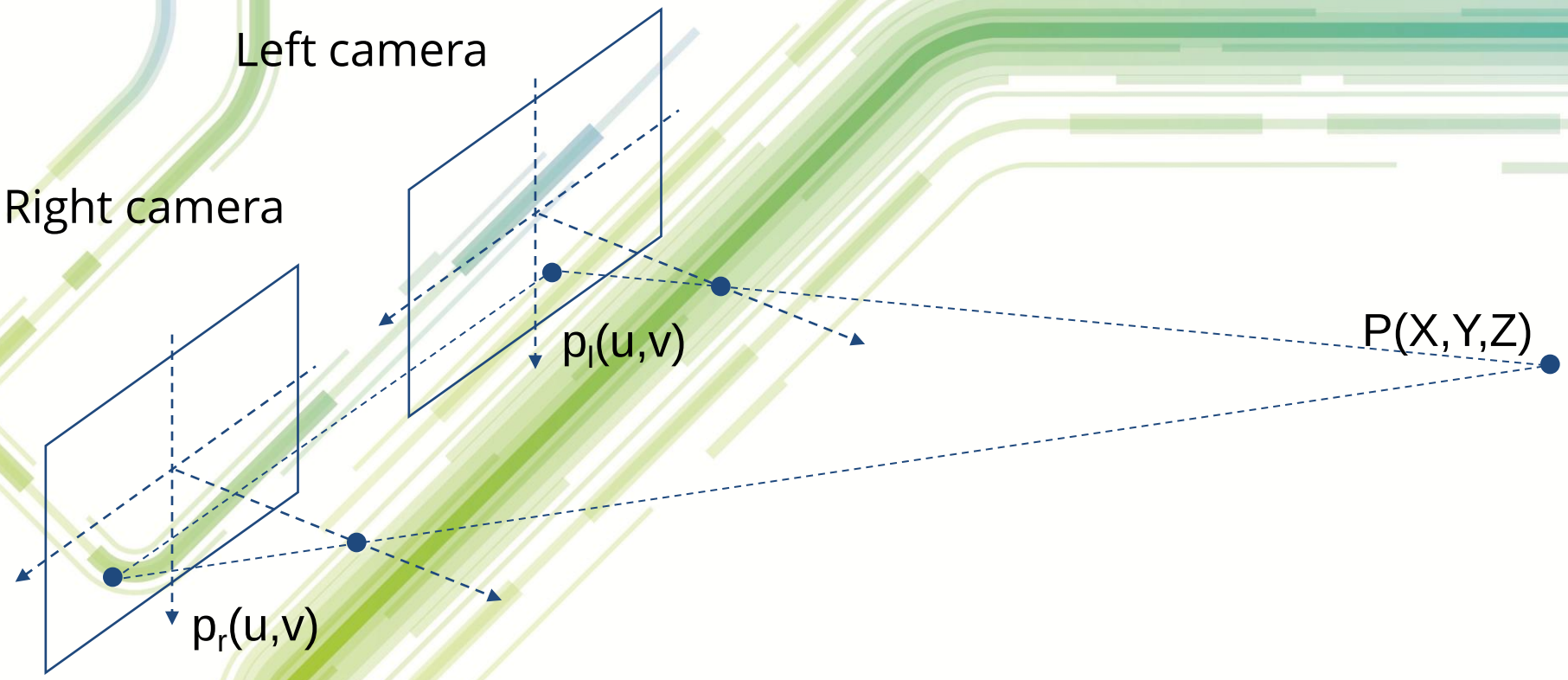


**III.**

Based on 2D images recover
3D structure of the scene



# Basic concept



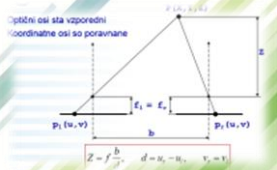

– It is possible to infer the depth from the stereo disparity (stereo parallax) $p_l(u,v) \neq p_r(u,v)$

# Why stereo?

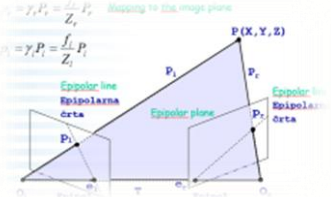
- It is biologically motivated
- Extensively studied and fairly well understood
- Solutions and technology exist
- Many needs and  uses in
  - –manufacturing/quality control,
  - –entertainment, …

# Stereo lecture outline

- Basic stereo vision model (lateral stereo model)



- Epipolar geometry, epipolar lines

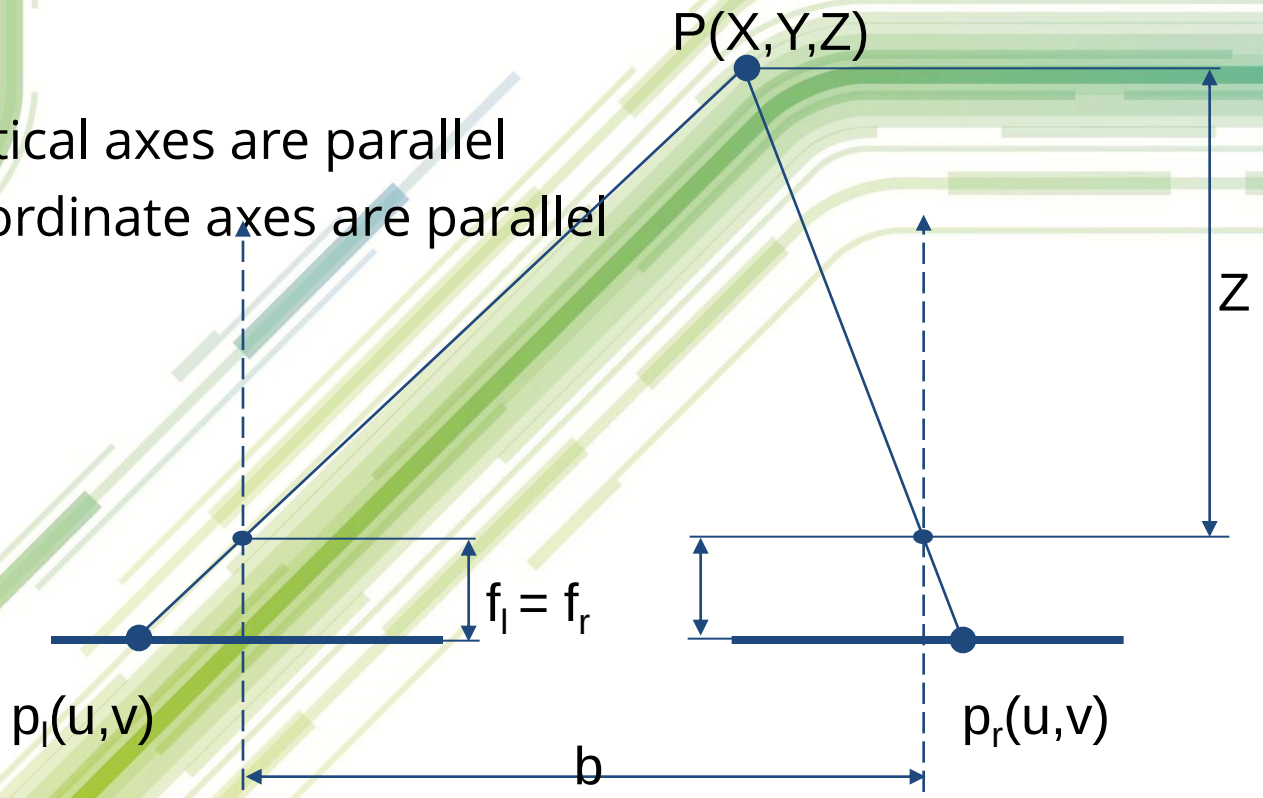



- Essential matrix and Fundamental matrix

- Stereo matching

# Lateral stereo model

- Optical axes are parallel
- Coordinate axes are parallel

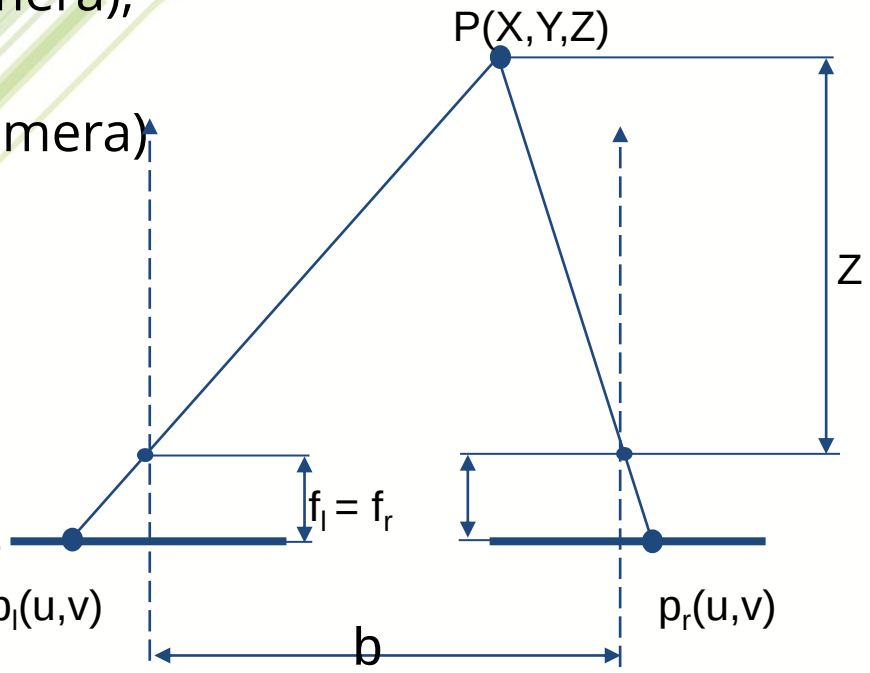



$$Z = f\frac{b}{d}, \qquad d = u_r - u_l, \qquad v_r = v_l$$

d = disparity!

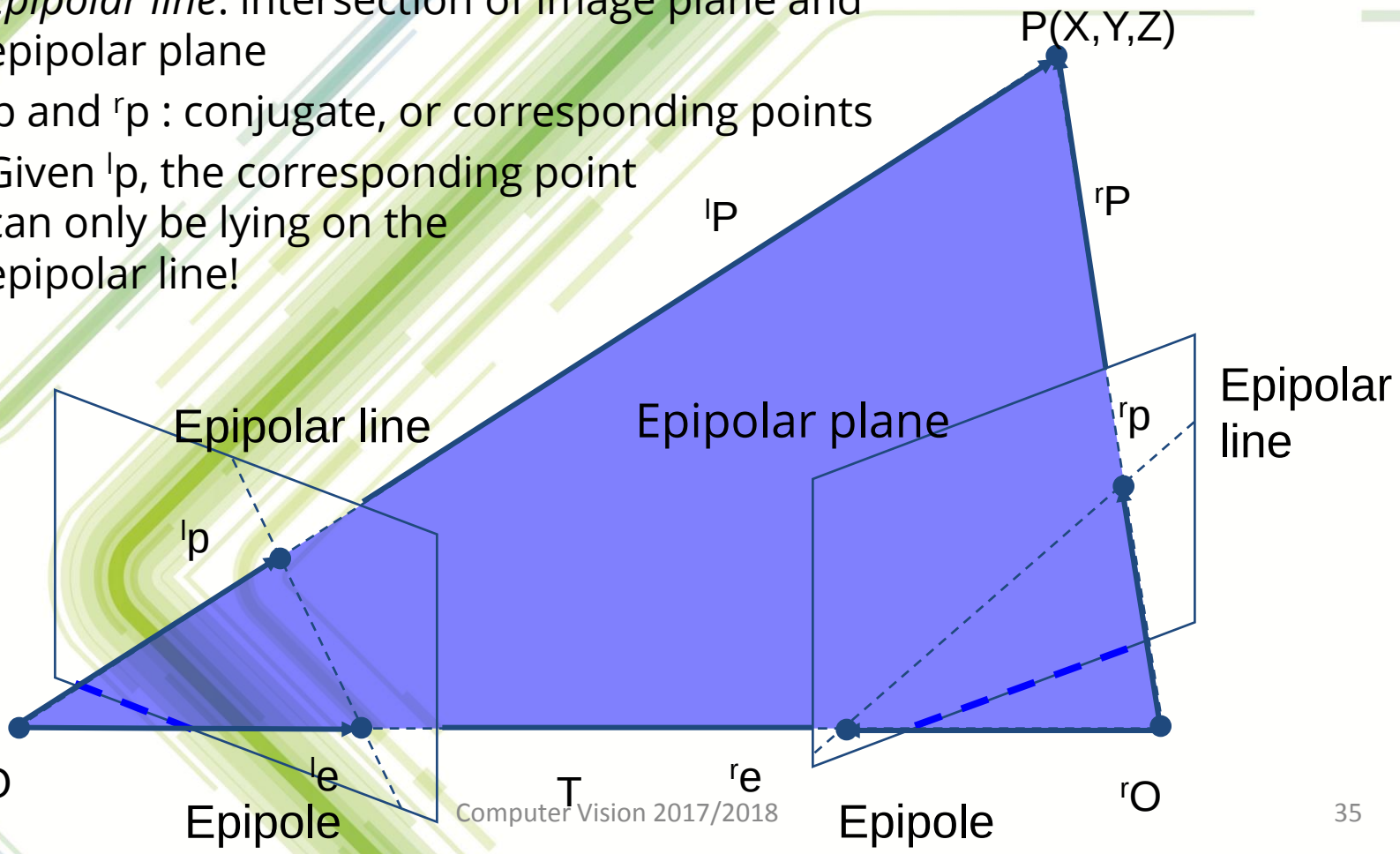
# Basic parameters of the stereo setup

- Basic parameters
  - Focal lengths (left and right camera), preferably equal
  - Image centers (left and right camera)
  - Pixel size
  - Baseline distance
- In general
  - Intrinsic (internal) parameters:
    - Mapping from camera coordinates to image coordinates: focal lengths, image centers, pixel size
  - Extrinsic (external) parameters:
    - mapping between camera coordinate systems (relative position and orientation)



# General case – nonparallel cameras

- *Epipolar plane*: defined by points P, $^{l}O$ and $^{r}O$
- *Epipolar line*: intersection of image plane and epipolar plane
- $^{l}p$ and $^{r}p$ : conjugate, or corresponding points
- Given $^{l}p$, the corresponding point can only be lying on the epipolar line!



# General case – nonparallel cameras

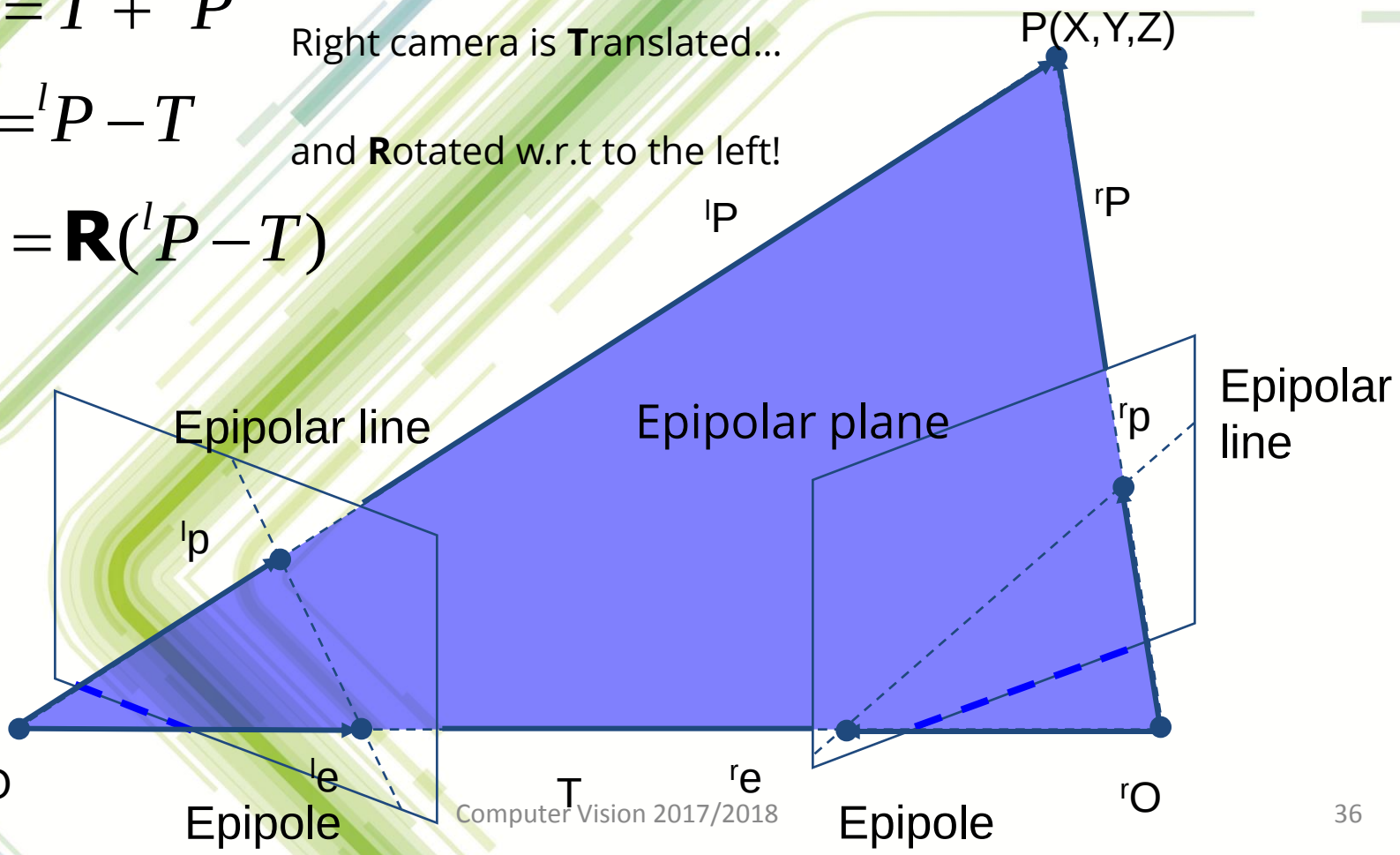
$$^{l}P = T + {}^{r}P$$

$$^{r}P = {}^{l}P - T$$

$$^{r}P = \mathbf{R}({}^{l}P - T)$$

Right camera is **T**ranslated…

and **R**otated w.r.t to the left!



# General case – nonparallel cameras

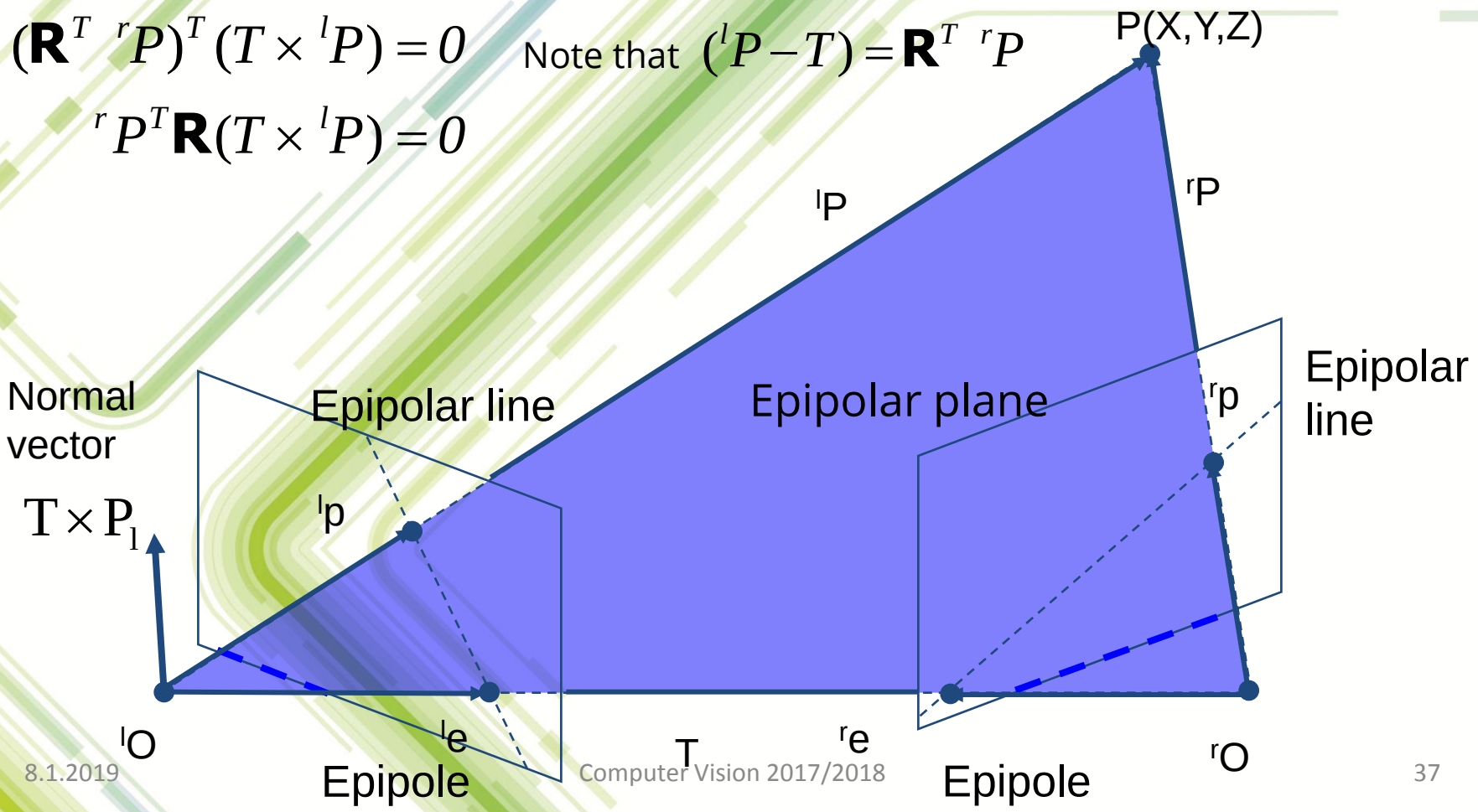
$$({}^{l}P - T)^{T}(T \times {}^{l}P) = 0$$

Epipolar plane equation!

$$(\mathbf{R}^{T}\,{}^{r}P)^{T}(T \times {}^{l}P) = 0$$

Note that $({}^{l}P - T) = \mathbf{R}^{T}\,{}^{r}P$

$${}^{r}P^{T}\mathbf{R}(T \times {}^{l}P) = 0$$



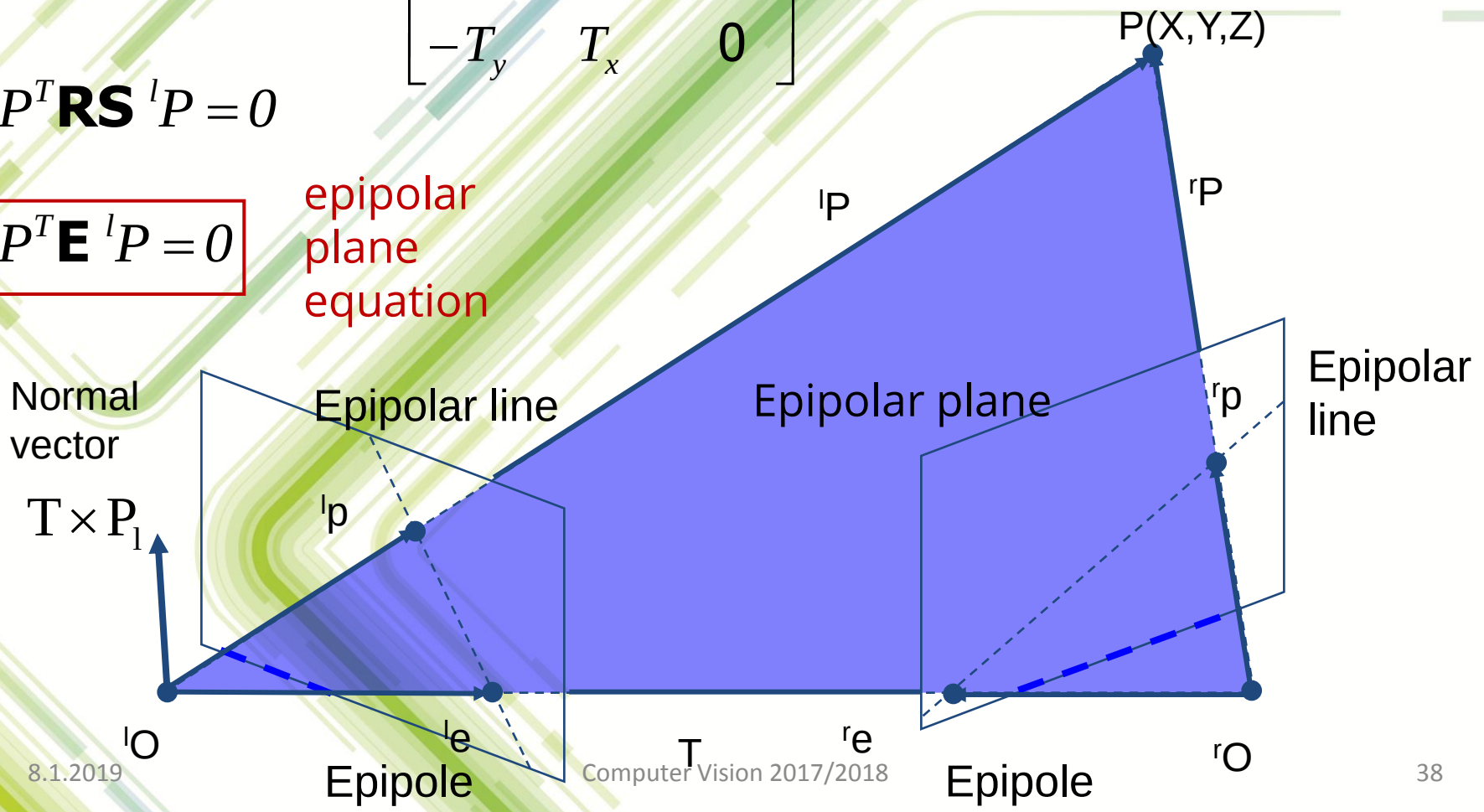
# General case – nonparallel cameras

$$\text{Op.}: T \times {}^{l}P = \mathbf{S}\ {}^{l}P = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} {}^{l}P$$

$${}^{r}P^{T}\mathbf{RS}\ {}^{l}P = 0$$

$${}^{r}P^{T}\mathbf{E}\ {}^{l}P = 0$$

epipolar plane equation



# Essential matrix

$$ {}^{r}P^{T}\mathbf{E}{}^{l}P = 0 $$

${}^{r}P$ and ${}^{l}P$ represent the same scene point P w.r.t left and right camera coordinate systems

$$ \mathbf{E} = \mathbf{RS} $$

- External (extrinsic) parameters
- E: Essential matrix, (3 x 3) of rank 2
  - S: translation matrix (rank 2)
  - R: rotation matrix (rank 3)
- E is singular, has two equal non-zero singular values and one zero singular value

# Essential matrix

$$ {}^{r}P^{T}\mathbf{E}\,{}^{l}P = 0 \qquad \mathbf{E} = \mathbf{RS} $$

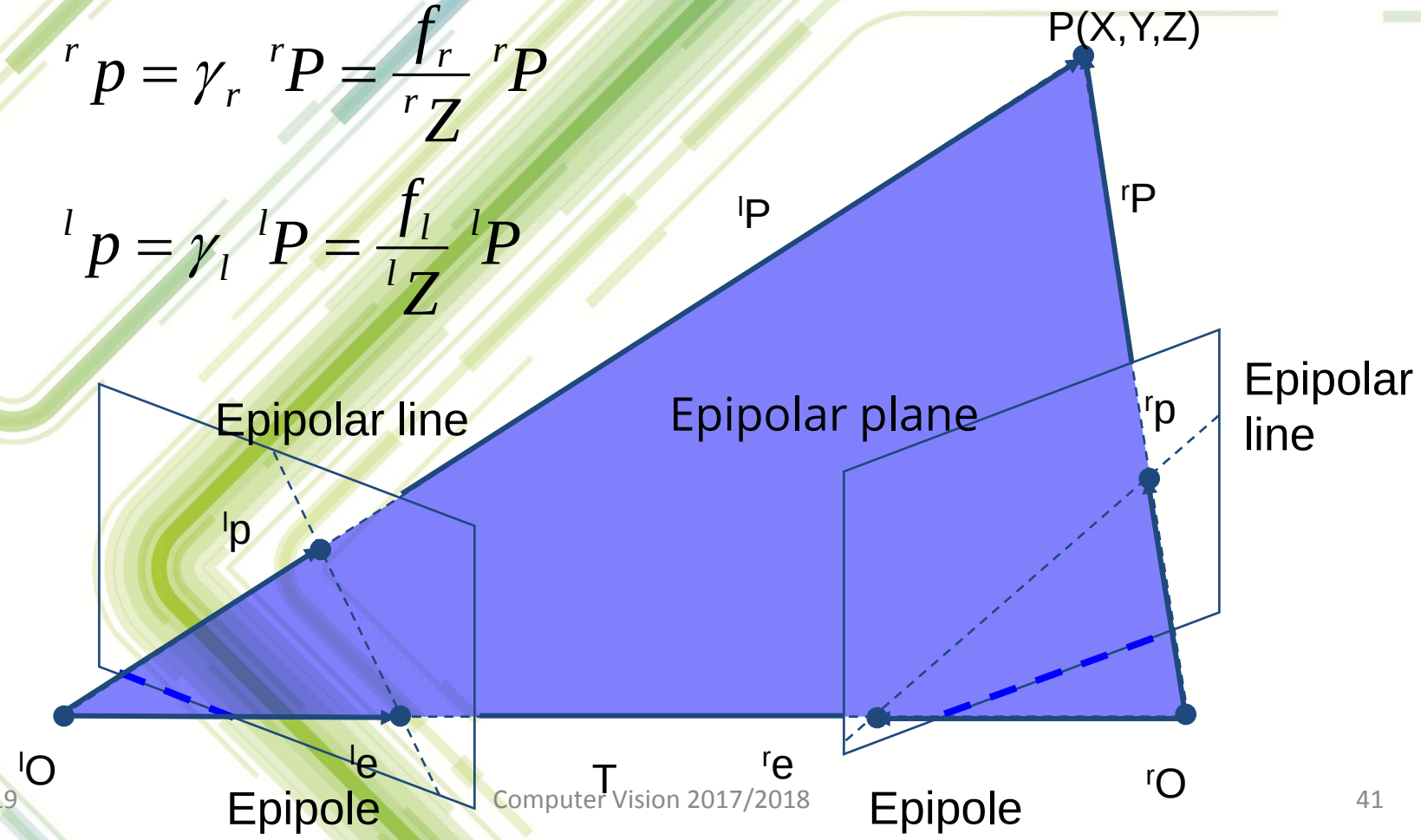
- Note,
  - given essential matrix E and point position ${}^{l}P$ in the left camera, point position ${}^{r}P$ in the right camera (or the oposite), can be determined up to a scale factor!
  - R has three degrees of freedom (3 angles of rotation)
  - S depends on three translation components, Tx, Ty, Tz, but determined only up to a scale, thus T = [Tx, Ty, 1].
  - That leaves 2 DOF
  - Therefore, E has 3+2 = 5 DOF

# Epipolar geometry

- Mapping to the image plane

$$ {}^{r}p = \gamma_r \; {}^{r}P = \frac{f_r}{{}^{r}Z} \; {}^{r}P $$

$$ {}^{l}p = \gamma_l \; {}^{l}P = \frac{f_l}{{}^{l}Z} \; {}^{l}P $$



# Fundamental matrix

Mapping to the image plane

$${}^{r}p = \gamma_r \, {}^{r}P = \frac{f_r}{{}^{r}Z} \, {}^{r}P$$

$${}^{l}p = \gamma_l \, {}^{l}P = \frac{f_l}{{}^{l}Z} \, {}^{l}P$$

${}^{l}p$ and ${}^{r}p$ are in real coordinates

K: internal (intrinsic) parameters

$${}^{r}p = \mathbf{K}_r \, {}^{r}P \quad \Rightarrow \quad {}^{r}P = \mathbf{K}_r^{-1} \, {}^{r}p$$

$${}^{l}p = \mathbf{K}_l \, {}^{l}P \quad \Rightarrow \quad {}^{l}P = \mathbf{K}_l^{-1} \, {}^{l}p$$

${}^{l}p$ and ${}^{r}p$ in homogeneous coordinates!

$${}^{r}P^{T} \, \mathbf{E} \, {}^{l}P = 0$$

$${}^{r}p^{T} \mathbf{K}_r^{-T} \mathbf{E} \mathbf{K}_l^{-1} \, {}^{l}p = 0$$

F: *fundamental* matrix

$${}^{r}p^{T} \mathbf{F} \, {}^{l}p = 0$$

# Fundamental matrix

$$ {}^{r}p^{T}\mathbf{F}\,{}^{l}p = 0 $$

$$ \mathbf{F} = \mathbf{K}_{r}^{-T}\mathbf{E}\mathbf{K}_{l}^{-1} $$

$$ {}^{r}l = \mathbf{F}\,{}^{l}p $$

$$ {}^{r}p^{T}\,{}^{r}l = 0 $$

- F: fundamental matrix, 3x3 of rank 2
- F depends on internal and external parameters, has 7 DOF
- l is a vector of line coefficients
- Line equation, i.e., epipolar line equation

$$ p_r^T\mathbf{F}p_l = ({}^{r}p^{T}\mathbf{F}\,{}^{l}p)^T = {}^{l}p^{T}({}^{r}p^{T}\mathbf{F})^T = {}^{l}p^{T}\mathbf{F}^T\,{}^{r}p = {}^{l}p^{T}\,{}^{l}l = 0 $$

- Given point in one image, by knowing F, we can determine the corresponding epipolar line in the second image!
  - Finding stereo *correspondences* is much easier this way!

# E, notations and conventions

Derivation of E in previous slides directly followed T.V. book.

$${}^{r}P^{T}\mathbf{E}\,{}^{l}P=0 \quad \text{with} \quad \mathbf{E}=\mathbf{RS}$$

where S is defined with respect to the left camera, and R is defined with respect to the right camera.

More often, however, you encounter different, though equivalent definition of E,

$${}^{r}P^{T}\mathbf{E}\,{}^{l}P=0 \quad \text{with} \quad \mathbf{E}=\mathbf{SR}$$

wheras R and S now define a “pose” of the left camera with respect to the right camera.

Also note: $({}^{r}P^{T}\mathbf{E}\,{}^{l}P)^{T}=(\mathbf{E}\,{}^{l}P)^{T}\,{}^{r}P={}^{l}P^{T}\mathbf{E}^{T}\,{}^{r}P=0$

# E, notations and conventions

Frequently, epipolar constraint equation is written like

$$ {}^{r}\tilde{p}^{T}\mathbf{E}\,{}^{l}\tilde{p}=0 $$

where lower p stands for left and right image coordinates, respectively (in homogeneous form)

In this case we assume ‚canonical' camera with f = 1 and image origin aligned with the principal point. Thus, internal parameter matrix is identity matrix.

$$ \mathbf{K}=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} $$

In other words, we assume that internal parameters are known.

# Derivation of E, algrebraic approach

Projection matrices for the two cameras, note that K = I, and $\left[{}^{W}\mathbf{R}_{l}|{}^{W}\mathbf{t}_{l}\right]$ and $\left[{}^{W}\mathbf{R}_{r}|{}^{W}\mathbf{t}_{r}\right]$ define poses of cameras wrt world (W)

$${}^{l}\gamma\ {}^{l}\tilde{p}={}^{l}\mathbf{K}\left[{}^{W}\mathbf{R}_{l}^{T}|-{}^{W}\mathbf{R}_{l}^{T}{}^{W}\mathbf{t}_{l}\right]{}^{w}P=\mathbf{I}\left[\mathbf{I}|\mathbf{0}\right]{}^{w}\tilde{P}$$

$${}^{r}\gamma\ {}^{r}\tilde{p}={}^{r}\mathbf{K}\left[{}^{W}\mathbf{R}_{r}^{T}|-{}^{W}\mathbf{R}_{r}^{T}{}^{W}\mathbf{t}_{r}\right]{}^{w}P=\mathbf{I}\left[{}^{W}\mathbf{R}_{r}^{T}|-{}^{W}\mathbf{R}_{r}^{T}{}^{W}\mathbf{t}_{r}\right]{}^{w}\tilde{P}$$

Because $W\equiv l$

$$\left[{}^{W}\mathbf{R}_{r}^{T}|-{}^{W}\mathbf{R}_{r}^{T}{}^{W}\mathbf{t}_{r}\right]=\left[{}^{l}\mathbf{R}_{r}^{T}|-{}^{l}\mathbf{R}_{r}^{T}{}^{l}\mathbf{t}_{r}\right]=\left[{}^{r}\mathbf{R}_{l}|{}^{r}\mathbf{t}_{l}\right]$$

This is the pose of left camera wrt right camera

# Derivation of E, algrebraic approach

Therefore

$$ {}^{r}\gamma\, {}^{r}\tilde{p} = \left[ {}^{r}\mathbf{R}_l \middle| {}^{r}\mathbf{t}_l \right] {}^{w}\tilde{P} = {}^{r}\mathbf{R}_l\, {}^{w}P + {}^{r}\mathbf{t}_l = {}^{r}\mathbf{R}_l\, {}^{l}\gamma\, {}^{l}\tilde{p} + {}^{r}\mathbf{t}_l \quad /\times {}^{r}\mathbf{t}_l $$

Note that ${}^{r}\mathbf{t}_l \times {}^{r}\mathbf{t}_l = 0$

$$ {}^{r}\gamma\, {}^{r}\mathbf{t}_l \times {}^{r}\tilde{p} = {}^{r}\mathbf{t}_l \times {}^{r}\mathbf{R}_l\, {}^{l}\gamma\, {}^{l}\tilde{p} + 0 \quad /\cdot\, {}^{r}\tilde{p} $$

$$ {}^{r}\gamma\, {}^{r}\tilde{p}^{T} \cdot {}^{r}\mathbf{t}_l \times {}^{r}\tilde{p} = {}^{r}\tilde{p}^{T} \cdot {}^{r}\mathbf{t}_l \times {}^{r}\mathbf{R}_l\, {}^{l}\gamma\, {}^{l}\tilde{p} $$

0

$$ {}^{r}\tilde{p}^{T} \cdot {}^{r}\mathbf{t}_l \times {}^{r}\mathbf{R}_l\, {}^{l}\gamma\, {}^{l}\tilde{p} = {}^{r}\tilde{p}^{T} \cdot {}^{r}\mathbf{S}_l \cdot {}^{r}\mathbf{R}_l\, {}^{l}\gamma\, {}^{l}\tilde{p} $$

Note that $\mathbf{t} \times \mathbf{v} = \mathbf{S}\mathbf{v}$

$$ {}^{r}\tilde{p}^{T}\, \mathbf{E}\, {}^{l}\tilde{p} = 0 $$

Note that $\mathbf{E} = \mathbf{S}\mathbf{R}$

# Simple (synthetic) example

*Camera 1 parameters* (‚left' camera):
f = 8 mm, pixel size = 10 μm, image size: 1280 x 960 (pixels)
image center: (640,480), no lens distortion
translation: (-0.1m, 0.0, 0.0), rotation about y (vertical) axis: 8 deg

*Camera 2 parameters* (‚right' camera), same intrinsic parameters:
f = 8 mm, pixel size = 10 μm, image size: 1280 x 960 (pixels)
image center: (640,480), no lens distortion
translation: (0.1m, 0.0, 0.0), rotation about y (vertical) axis: -8 deg

In other words: the same camera
Once rotated ‚right' for 8 deg. And translated ‚left' for 10cm,
Once rotated ‚left' for 8 deg. And translated ‚right' for 10cm.

# Simple (synthetic) example

Left Camera (intrinsic) $K_l$:

| | | |
|---|---|---|
| 800 | 0 | 640 |
| 0 | 800 | 480 |
| 0 | 0 | 1 |

Right Camera (intrinsic) $K_r$:

| | | |
|---|---|---|
| 800 | 0 | 640 |
| 0 | 800 | 480 |
| 0 | 0 | 1 |

Left Camera pose (extrinsic) $C_l$:

| | | | |
|---|---|---|---|
| 0.9903 | 0 | 0.1392 | -0.1000 |
| 0 | 1.0000 | 0 | 0 |
| -0.1392 | 0 | 09903 | 0 |
| 0 | 0 | 0 | 1.0000 |

Right Camera pose (extrinsic) $C_r$:

| | | | |
|---|---|---|---|
| 0.9903 | 0 | -0.1392 | 0.1000 |
| 0 | 1.0000 | 0 | 0 |
| 0.1392 | 0 | 0.9903 | 0 |
| 0 | 0 | 0 | 1.0000 |

Left Camera projection matrix $M_l$:

| | | | |
|---|---|---|---|
| 881.2852 | 0 | 522.4331 | 88.1285 |
| 66.8031 | 800.0 | 475.3287 | 6.6803 |
| 0.1392 | 0 | 0.9903 | 0.0139 |

Right Camera projection matrix $M_r$:

| | | | |
|---|---|---|---|
| 703.1437 | 0 | 745.1100 | -70.3144 |
| -66.8031 | 800.0 | 475.3287 | 6.6803 |
| -0.1392 | 0 | 0.9903 | 0.0139 |

# Simple (synthetic) example

Take a single scene point wrt world coordinate system

$$P = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.08 \\ 0.15 \\ 0.40 \end{bmatrix}$$

Transform it **to the left camera** coordinate system,

$$\widetilde{P}_l = C_l^{-1} \begin{bmatrix} P \\ 1 \end{bmatrix} = \begin{bmatrix} 0.9903 & 0 & -0.1392 & 0.0990 \\ 0 & 1 & 0 & 0 \\ 0.1392 & 0 & 0.9903 & 0.0139 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.08 \\ 0.15 \\ 0.04 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.1226 \\ 0.1500 \\ 0.4212 \\ 1 \end{bmatrix}$$

Project it into **the left image** coordinate system

$$\widetilde{p}_l = KP_l = \begin{bmatrix} 800 & 0 & 640 \\ 0 & 800 & 480 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.1226 \\ 0.1500 \\ 0.4212 \end{bmatrix} = \begin{bmatrix} 367.6046 \\ 322.1560 \\ 0.4212 \end{bmatrix} = \begin{bmatrix} 872.8416 \\ 764.9284 \\ 1 \end{bmatrix}$$

Point position in the left image $p_l$ = [873 765]

# Simple (synthetic) example

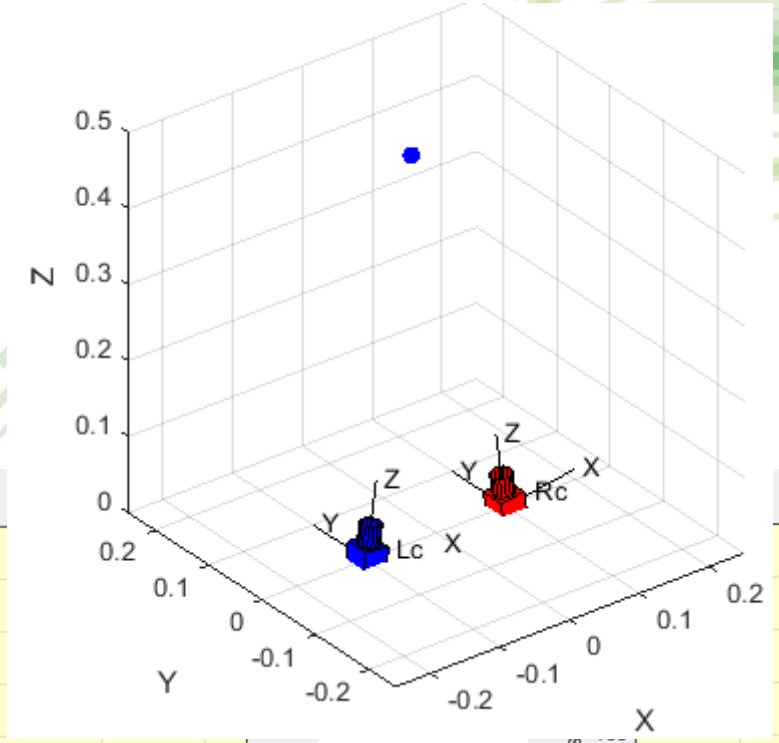
Of course, we could simply apply the camera projection matrix M,

$$ {}^{l}\tilde{p} = \mathbf{M}_{l}\tilde{P} = \begin{bmatrix} 881.2852 & 0 & 522.4331 & 88.1285 \\ 66.8031 & 800.0 & 475.3287 & 6.6803 \\ 0.1392 & 0 & 0.9903 & 0.0139 \end{bmatrix} \begin{bmatrix} 0.08 \\ 0.15 \\ 0.40 \\ 1 \end{bmatrix} = \begin{bmatrix} 367.6046 \\ 322.1560 \\ 0.4212 \end{bmatrix} = 0.4212 \begin{bmatrix} 872.84 \\ 764.93 \\ 1 \end{bmatrix} $$

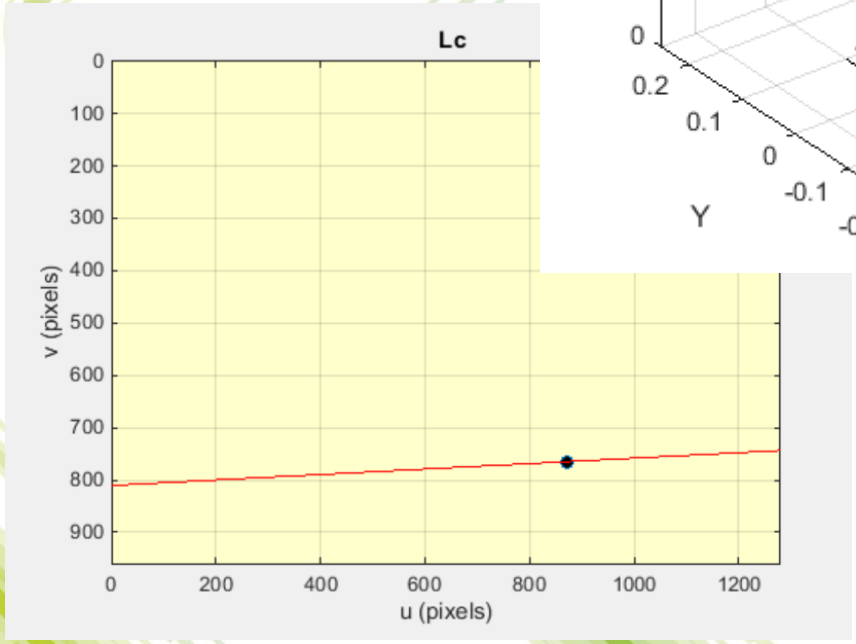
And the same for the right camera

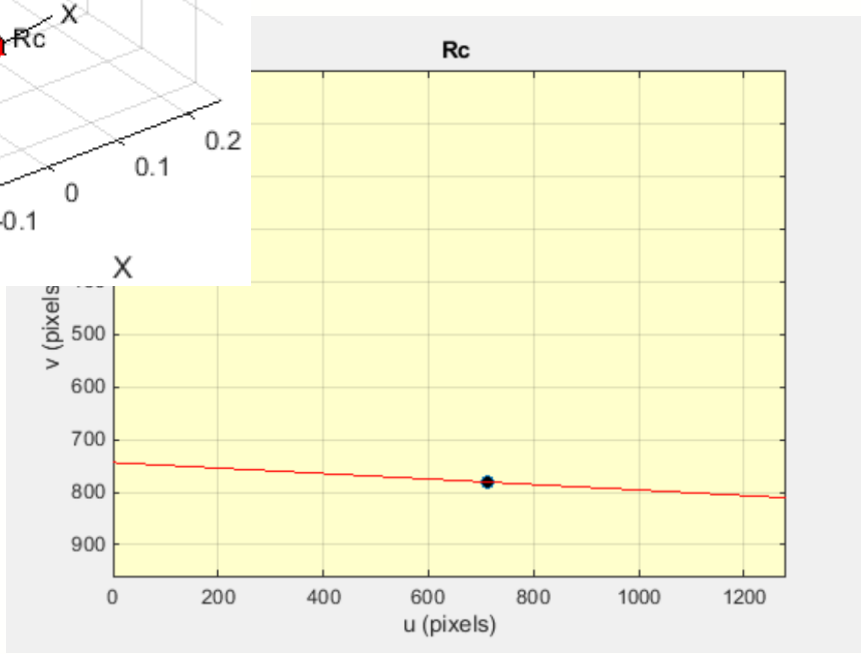
$$ {}^{r}\tilde{p} = \mathbf{M}_{r}\tilde{P} = \begin{bmatrix} 703.1437 & 0 & 745.1100 & -70.3144 \\ 66.8031 & 800.0 & 475.3287 & 6.6803 \\ 0.1392 & 0 & 0.9903 & 0.0139 \end{bmatrix} \begin{bmatrix} 0.08 \\ 0.15 \\ 0.40 \\ 1 \end{bmatrix} = \begin{bmatrix} 283.9811 \\ 311.4675 \\ 0.3989 \end{bmatrix} = 0.3989 \begin{bmatrix} 711.93 \\ 780.83 \\ 1 \end{bmatrix} $$

# Simple (synthetic) example




Produced using
P. Corke rvctools
for Matlab



# Simple (synthetic) example

Let's take left camera as a reference. Relative orientation of the right camera wrt the left camera is:

$$ {}^{l}C_r = C_l^{-1} C_r = \begin{bmatrix} 0.9613 & 0 & -0.2756 & 0.1981 \\ 0 & 1 & 0 & 0 \\ 0.2756 & 0 & 0.09613 & 0.0278 \\ 0 & 0 & 0 & 1 \end{bmatrix}, $$

$$ \text{with } R = \begin{bmatrix} 0.9613 & 0 & -0.2756 \\ 0 & 1 & 0 \\ 0.2756 & 0 & 0.09613 \end{bmatrix}, \; t = \begin{bmatrix} 0.1981 \\ 0 \\ 0.0278 \end{bmatrix}, \; S = \begin{bmatrix} 0 & -0.0278 & 0 \\ 0.0278 & 0 & -0.1981 \\ 0 & 0.1981 & 0 \end{bmatrix} $$

The essential matrix E is,

$$ E = R^T . S = \begin{bmatrix} 0 & -0.0278 & 0 \\ -0.0278 & 0 & 0.1981 \\ 0 & -0.1981 & 0 \end{bmatrix} $$

# Simple (synthetic) example

But, with the definition of E as E = S R,
we compute the pose of the first (left) camera wrt second (right) camera.

$$\begin{bmatrix} {}^{r}\mathbf{R}_{l} & {}^{r}\mathbf{t}_{l} \\ 0 & 1 \end{bmatrix} = \mathbf{C}_{r}^{-1}\mathbf{C}_{l} = \begin{bmatrix} 0.9903 & 0 & 0.1392 & -0.1 \\ 0 & 1.0000 & 0 & 0 \\ -0.1392 & 0 & 0.9903 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0.9903 & 0 & -0.1392 & 0.1 \\ 0 & 1.0000 & 0 & 0 \\ 0.1392 & 0 & 0.9903 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} {}^{r}\mathbf{R}_{l} & {}^{r}\mathbf{t}_{l} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.9613 & 0 & 0.2756 & -0.1981 \\ 0 & 1.0000 & 0 & 0 \\ -0.2756 & 0 & 0.9613 & 0.0278 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{S} = \begin{bmatrix} 0 & -0.0278 & 0 \\ 0.0278 & 0 & 0.1981 \\ 0 & -0.1981 & 0 \end{bmatrix}$$

$$\mathbf{E} = \mathbf{SR} = \begin{bmatrix} 0 & -0.0278 & 0 \\ 0.0278 & 0 & 0.1981 \\ 0 & -0.1981 & 0 \end{bmatrix} \begin{bmatrix} 0.9613 & 0 & 0.2756 \\ 0 & 1.0000 & 0 \\ -0.2756 & 0 & 0.9613 \end{bmatrix} = \begin{bmatrix} 0 & -0.0278 & 0 \\ -0.0278 & 0 & 0.1981 \\ 0 & -0.1981 & 0 \end{bmatrix}$$

This gives us the same result as before, as it should.

# Making the sense of it

- Why does epipolar geometry matter?
- It constrains the search for correspondences,
  - 1D search along a line instead of 2D search within image!
- Improves robustness,
  - false matches are less likely to occur, matches that do not satisfy epipolar constraint, are rejected.

# Stereo calibration

- Solving for F is non-trivial computational problem.

$$\mathbf{p}_r^T \mathbf{F} \mathbf{p}_l = 0$$

  - In principle, we need 8 scene points, i.e., 8 pairs of corresponding image points, but more than that are recommended.
  - Each scene point contributes a single (homogeneous) equation.
  - This is an optimization problem that can be solved using:
    - LS
    - SVD
    - RANSAC

# Stereo calibration

- 8 point algorithm (conceptually)
  - Construct system of N equations formed by N correspondences (N >=8)

$$\mathbf{A}\mathbf{x} = \mathbf{0}$$

  - A is N x 9 matrix, x is vector of unknowns (i.e., F is 3x3=9 unknowns)
  - Compute SVD of A,

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

  - The solution x is column of V for the smallest singular value in $\mathbf{\Sigma}$

# Stereo calibration

- 8 point algorithm (conceptually)
  - Construct system of N equations formed by N correspondences (N >=8)

$$\mathbf{A}\mathbf{x} = \mathbf{0}$$

  - A is N x 9 matrix, x is vector of unknowns (i.e., F is 3x3=9 unknowns)
  - Compute SVD of A,

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

  - The solution x is column of V for the smallest singular value in $\mathbf{\Sigma}$
  - (but this is not the end)

# Stereo calibration

- To enforce singularity of F, compute SVD of F,

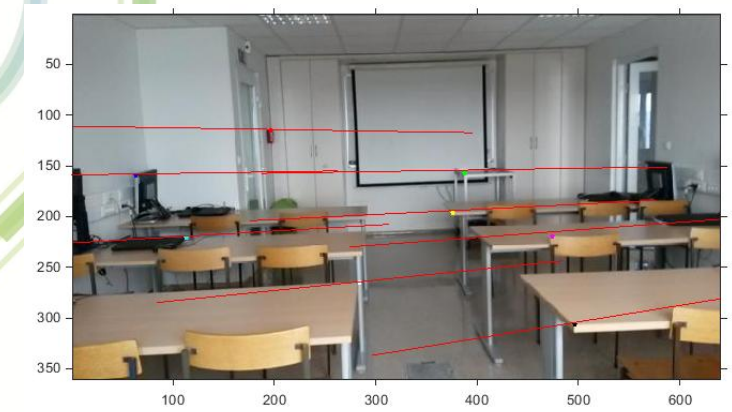
$$\mathbf{F} = \mathbf{U}_F D \mathbf{V}_F^T$$

- Set the smallest singular value in D to zero, D-> $D_c$, compute corrected F

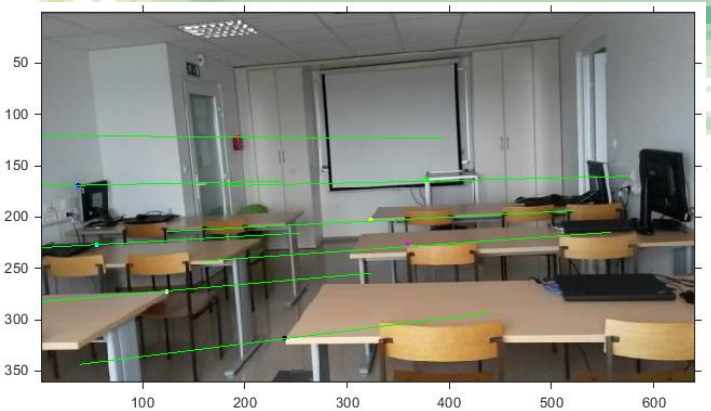
$$\mathbf{F} = \mathbf{U}_F D_C \mathbf{V}_F^T$$

# Simple example, estimate F

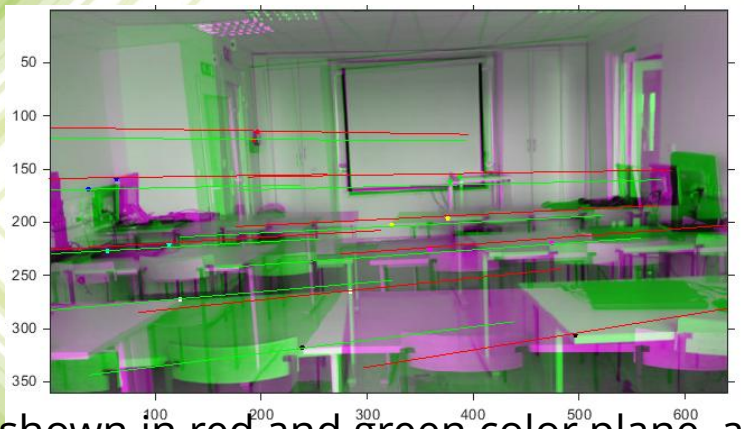
L




R



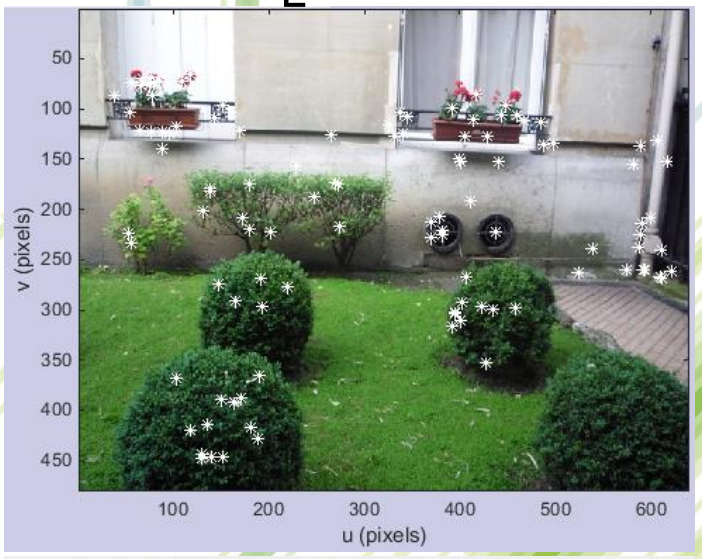

F was estimated from 8 correspondences (marked points), then epipolar lines were drawn for each point pair.



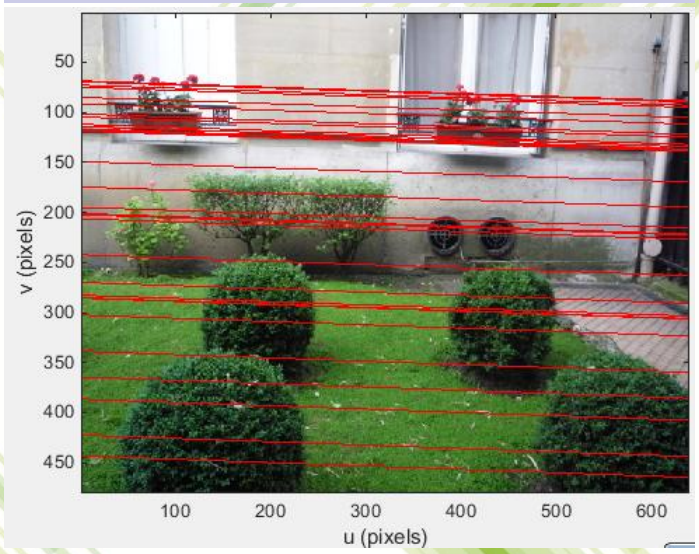

L and R image shown in red and green color plane, along with epipolar lines.

# Stereo calibration example

L




R

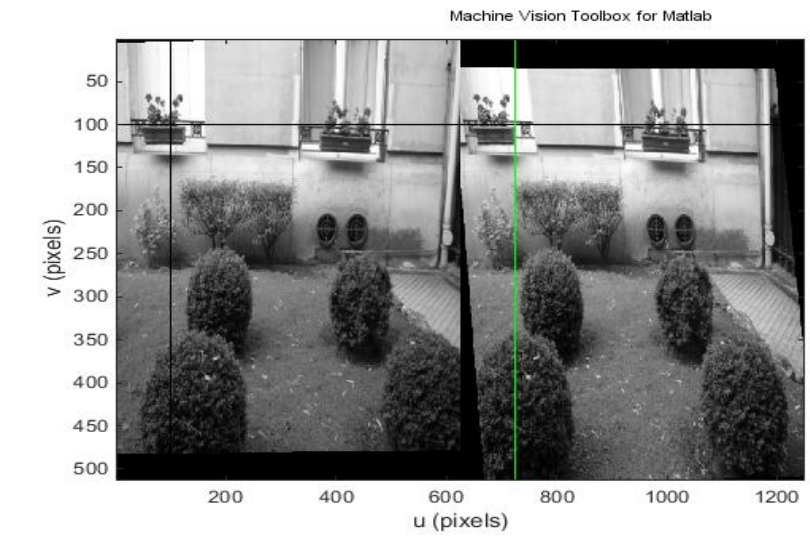




Key points detected
Key points matched
Fundamental matrix estimated, RANSAC
Epipolar lines drawn

Produced using
P. Corke rvctools for Matlab

# Rectification

- The task:
  - Vergent cameras (non-paralel) -> transform images to ‚ideal stereo pair'
  - Result: Horizontal lines are also epipolar lines. The images look like the images would have been taken using parallel cameras

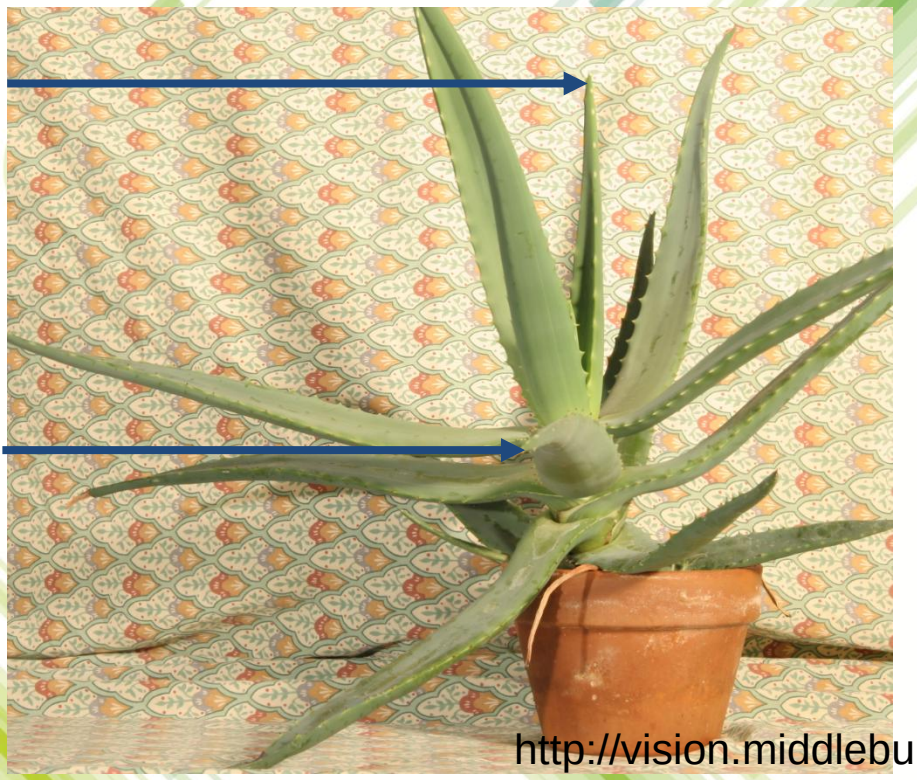


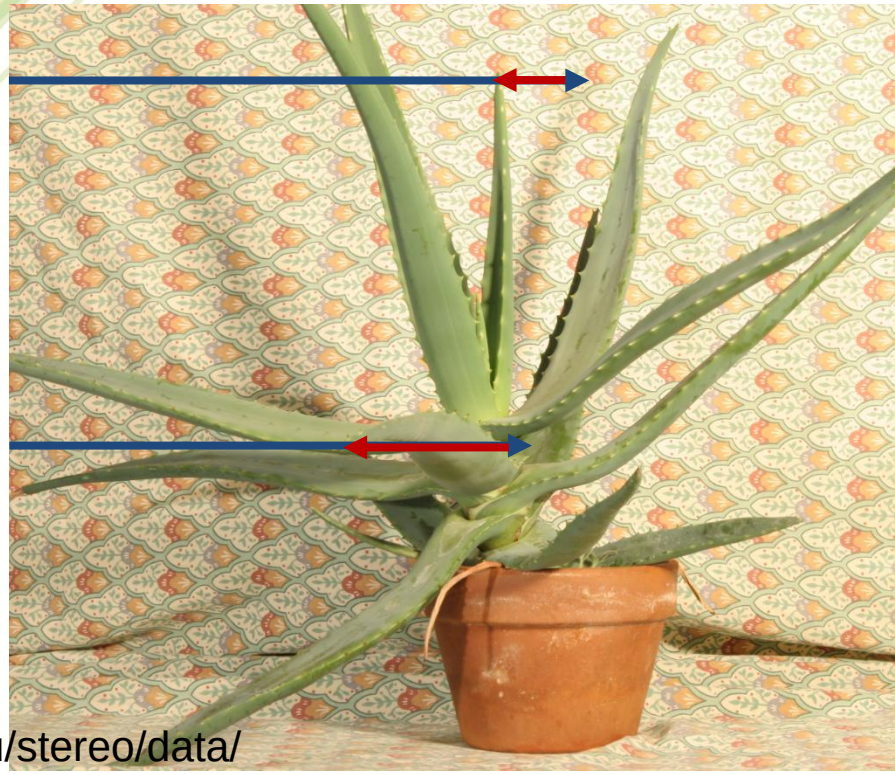

Produced using P. Corke rvctools for Matlab

# Stereo matching

# The goal: reconstruction

– Based on two slightly different images of the same scene, caused by camera translation, recover the ‚lost' (3rd) dimension (distance, range, depth).



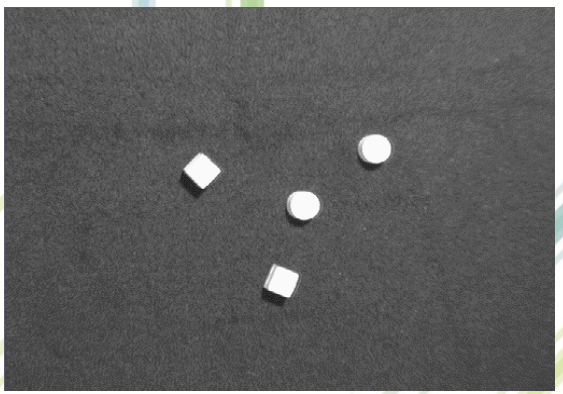


http://vision.middlebury.edu/stereo/data/

# The stereo problem

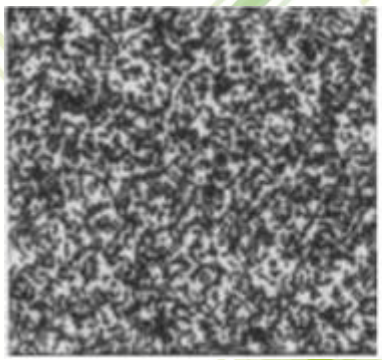
- Basic problems (“issues”) of the stereo vision:
  - Find correspondence, and therefore disparity, i.e., find points in the left and the right image that are projections of the same scene point.
  - The solution to this problem is called ‚*stereo matching*‘
  - Once the correspondence is solved, either for small or large number of points, compute 3D representation (‚structure‘) of the scene. That is called *reconstruction.*
  - Nevertheless, to do that, stereo *calibration* is needed.

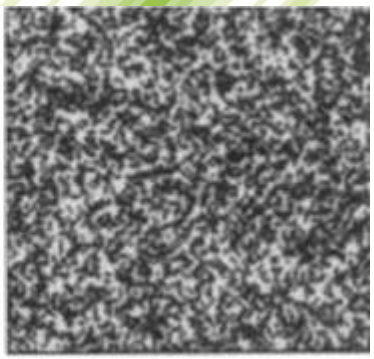
# Is stereo matching hard?

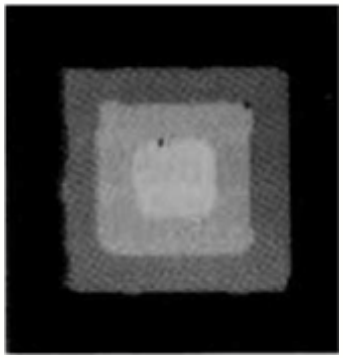


Trivial

L R







Non-trivial

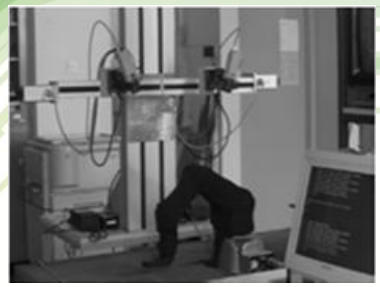
L R Disparity map / field

D. Marr 82: Vision

# Stereo cameras (rigs, heads)

TRC ~ 95
(active stereo head)

FE ~ 95



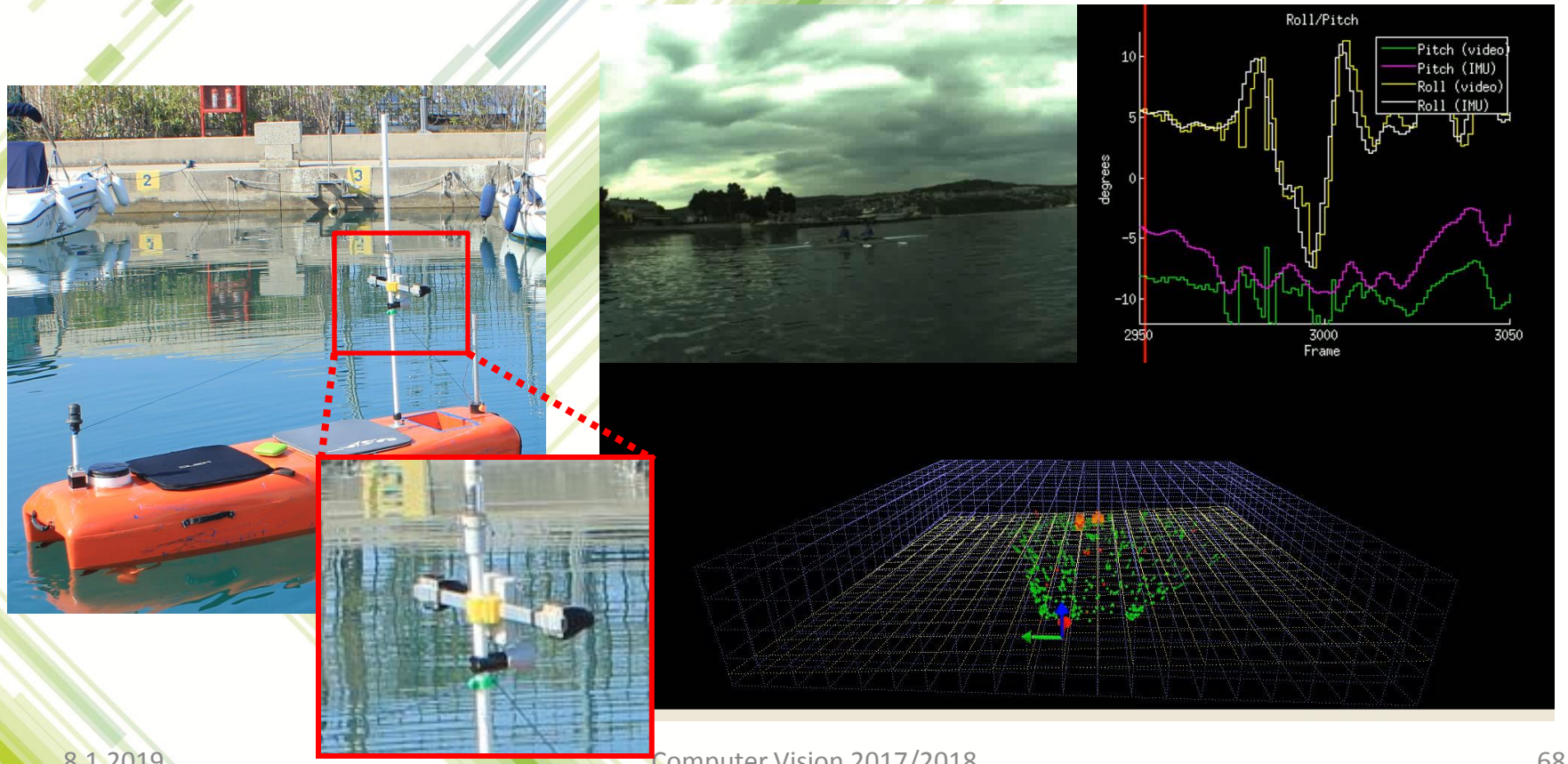
Point Grey
Bumblebee

IDS
(stereo+structured light)

Nao

# Stereo cameras (rigs, heads)

- FE LSV & Harpha Sea (2012)



# Stereo matching

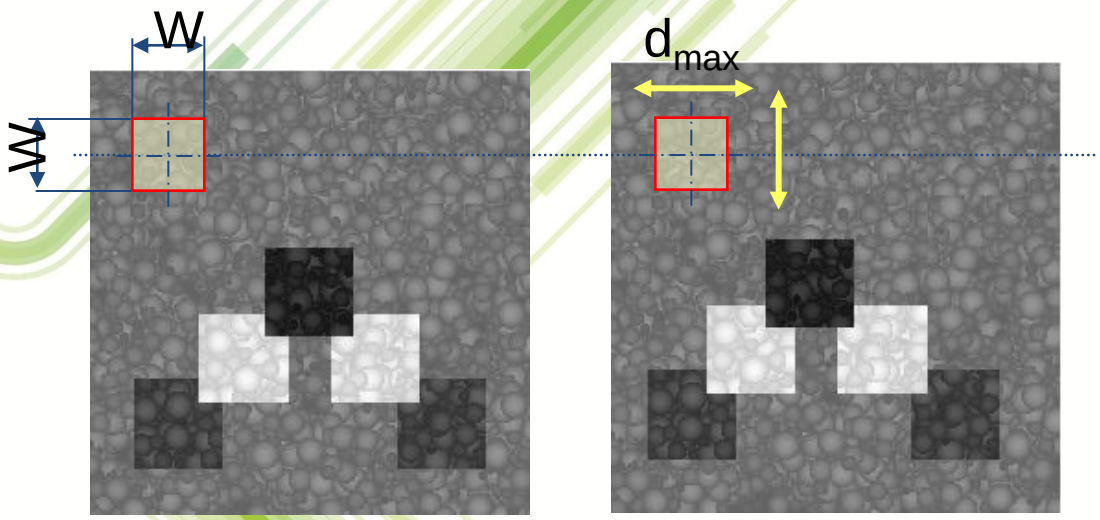
- For a point in one (left) image find the corresponding point in the other (right) image
- Once the correspondence is found, compute disparity
- Questions:
  - which points to match?
  - how?
- What is needed:
  - matching strategy
  - similarity measure

# Stereo matching strategies

- Stereo matching strategies:
  - Correlation based (region based, ‚dense' stereo).
    - Take a (square) patch in one image and search for a match in the other image.
    - Choose similarity measure (SM): SAD, SSD, NCC, ZNCC, …
    - search for match, find maximum, or minimum of SM
  - Feature based (‚local image features based')
    - detect key points (distinct image points), edges, corners, …
    - do that independently in both images
    - Perform key-point matching, search for matching point pairs,
    - using predefined distance/similarity measure.

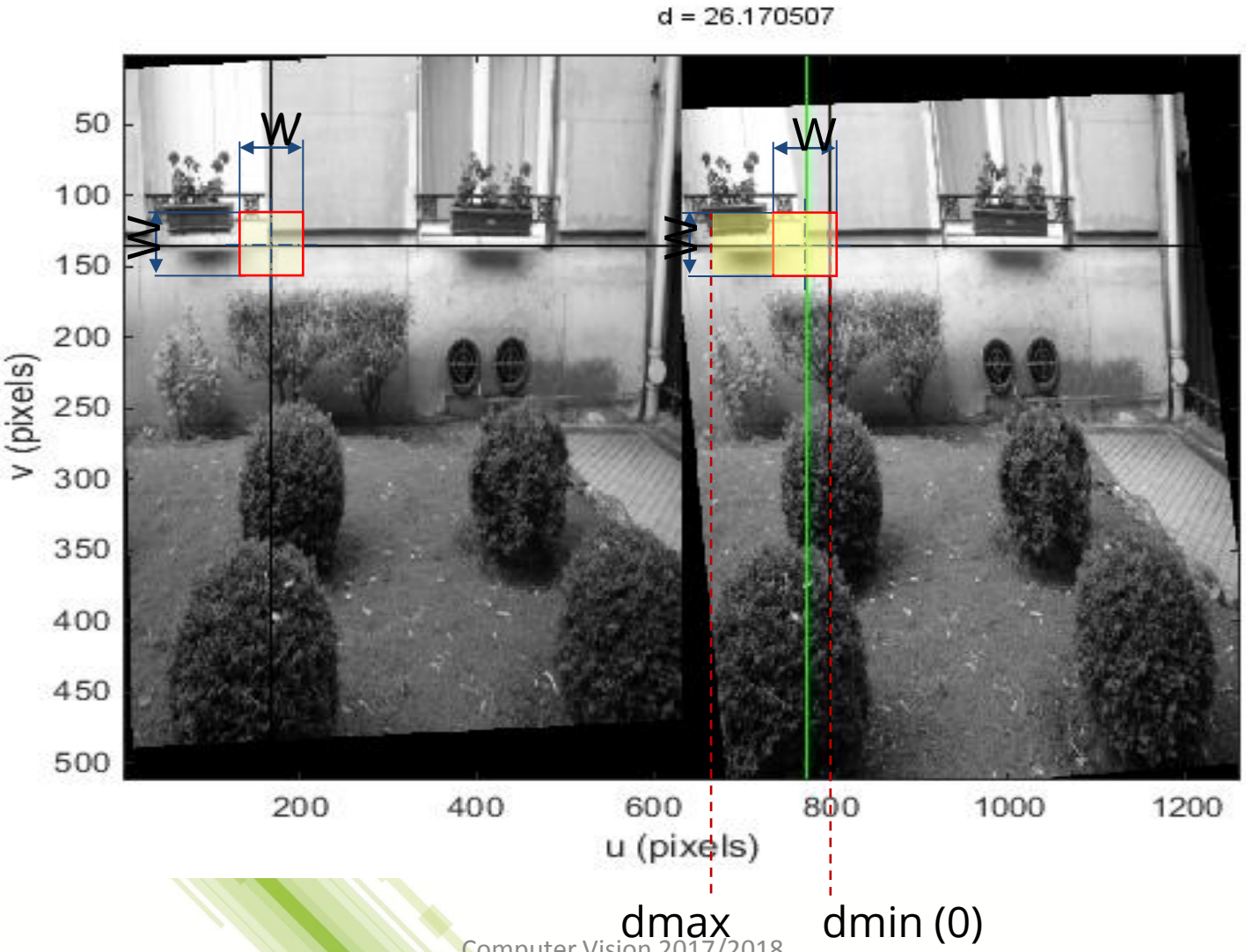
# Stereo matching

- Correlation based – what do we need?
  - select window size, the size of regions involved in matching
  - select search area, i.e., how far to search for correspondences
  - search for best match using predefined similarity measure




  - Note: synthetic images are quite convenient for the evaluation of matching algorithms.

# Stereo matching



# Stereo matching

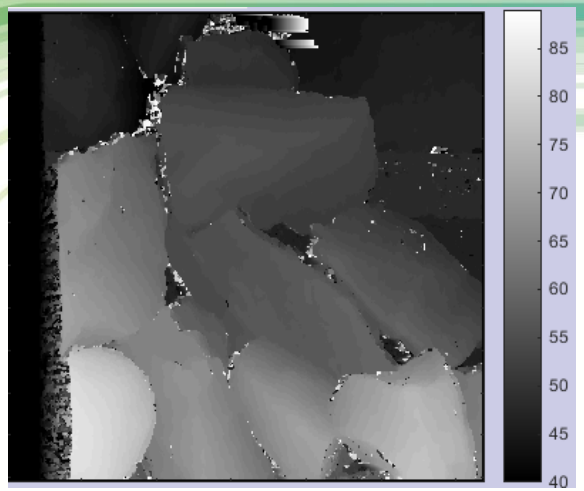
- Correlation based (we talked about that before)
  - Select similarity measure, S(d), d = (u,v):
    - Sum of absolute differences (within window), SAD,
    - Sum of square differences, SSD,
    - Correlation coeffiecient, CC,
    - Normalized correlation, NCC, ZNCC.
  - compute similarity measure as a funcion of displacement d, S(d),
  - search for the maximum

$$S(d(i,j)) = \sum_{k=-w}^{k=+w} \sum_{l=-w}^{l=+w} I_l(i+k, j+l) \bullet I_r(i+k+u, j+l+v)$$

$$d(i,j) = \arg\max_d \{S(d)\}$$

# Stereo matching example

L R



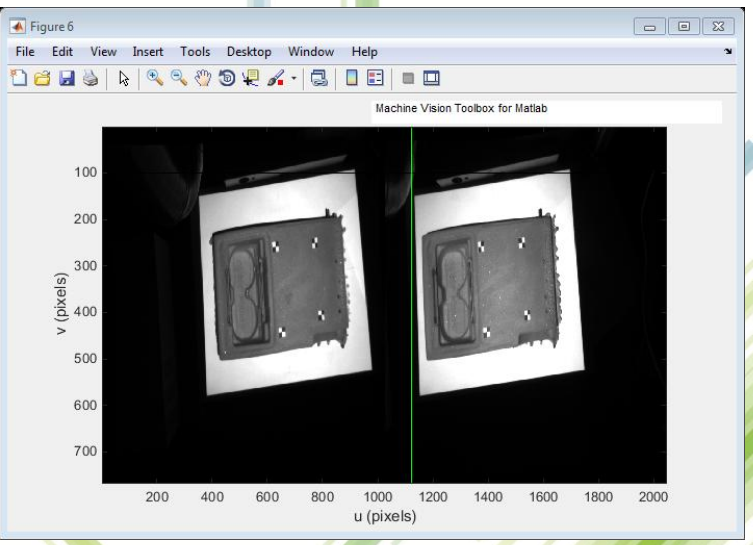

Similarity measure: ZNCC
Window size: 7 x 7

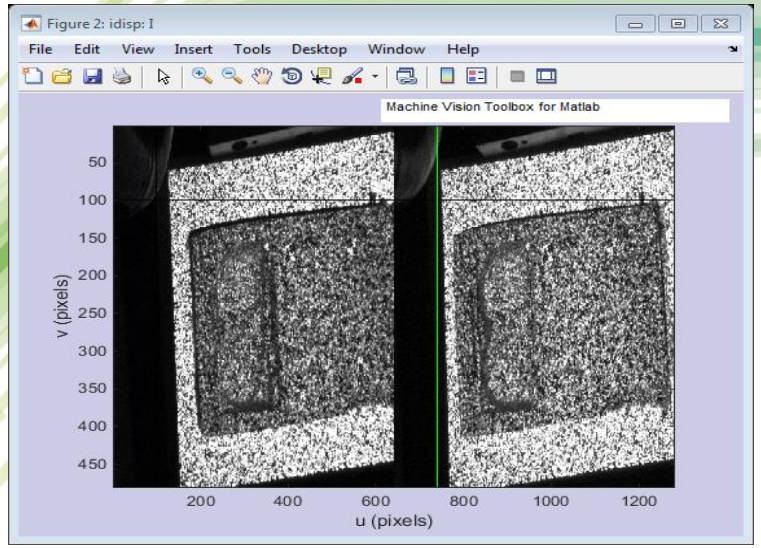
Produced using P. Corke RVCtools for Matlab

# Stereo matching

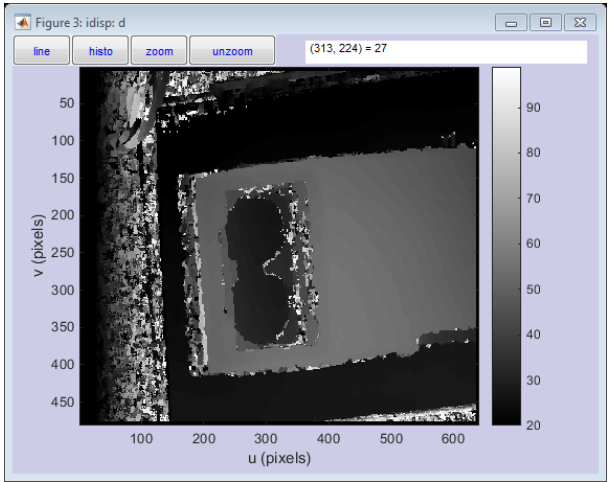
- Correlation based approaches:
  - produce dense disparity field (map), and therefore dense depth (+)
  - require textured surfaces
  - produce many false mathes (-)
- Note:
  - Regions not covered by both cameras cannot be matched
  - Regions hidden/self occluded in one image cannot be matched.

# Stereo matching example

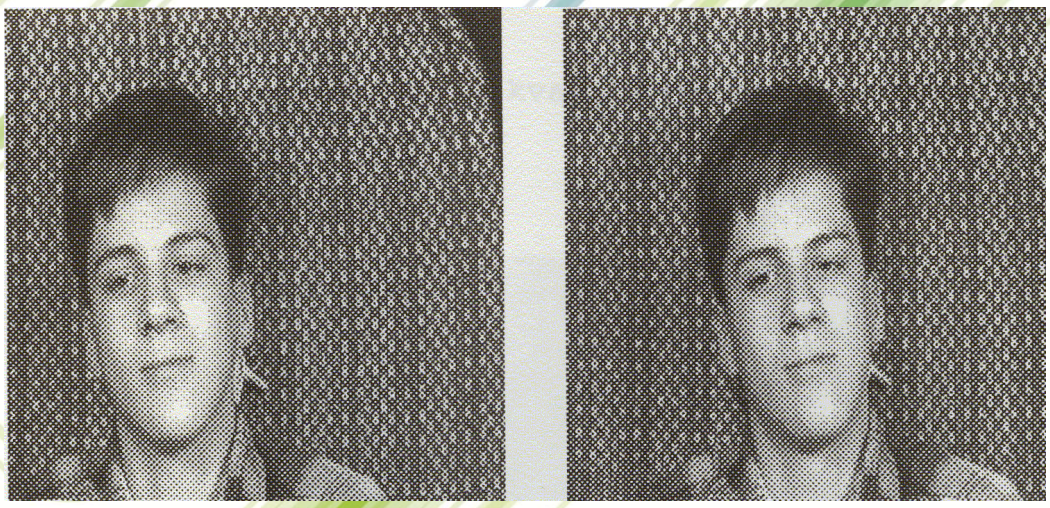


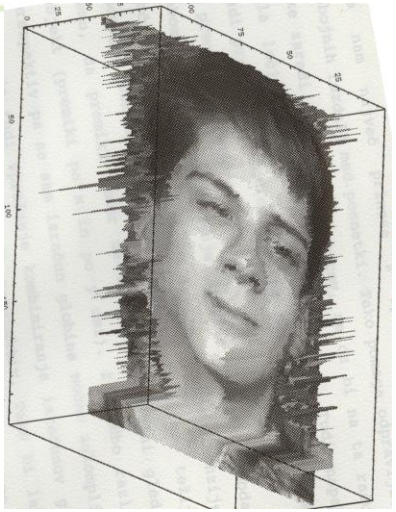




- Regions without texture (homogeneous regions) cannot be matched.
- Solution: *Stereo* + *Structured lighting*!
- But, occlusions are always problematic
- Furthermore, false matches due to
- ambiguity are likely to occur.



# Stereo example – a face



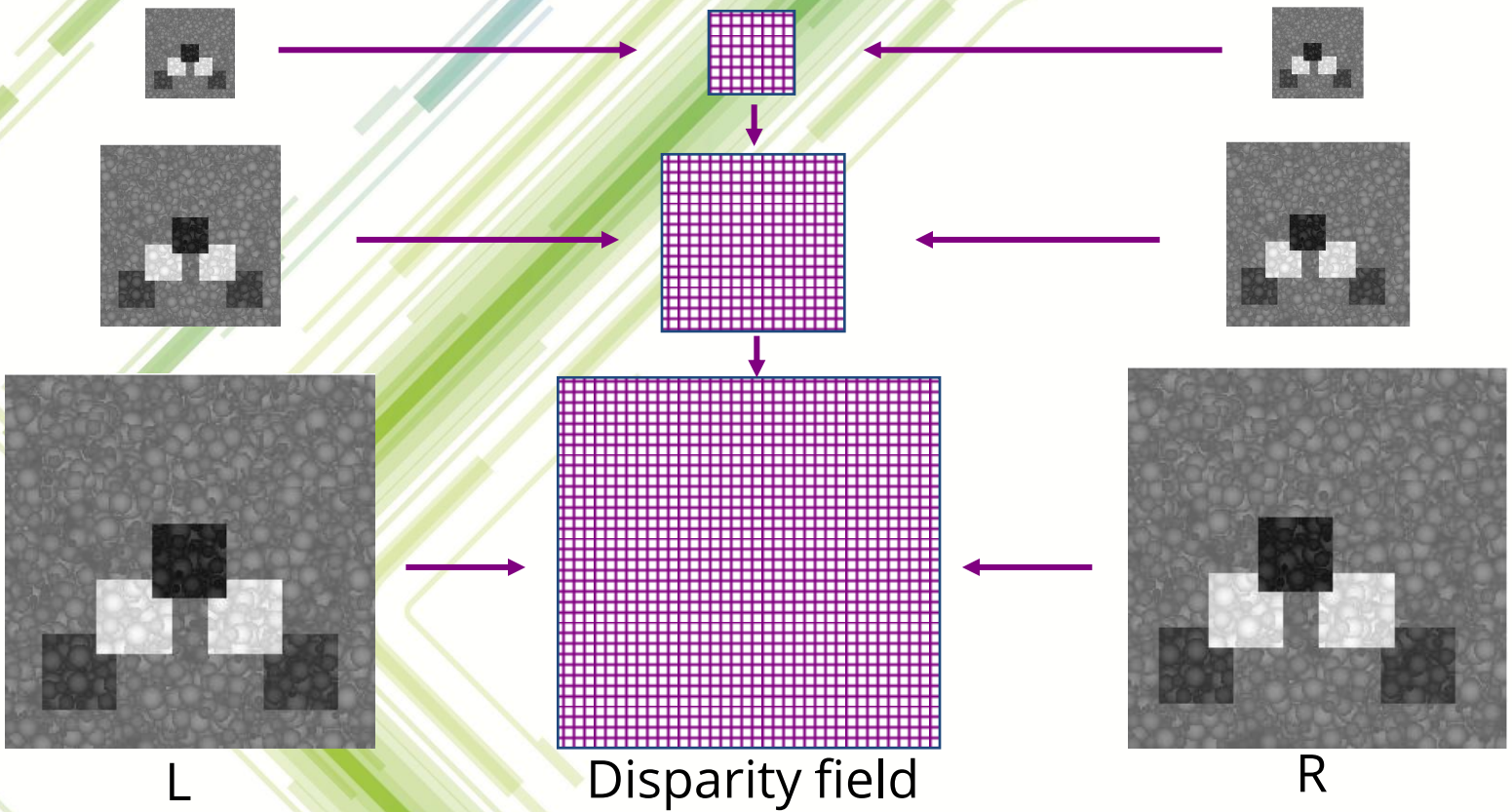


# Correlation based approaches

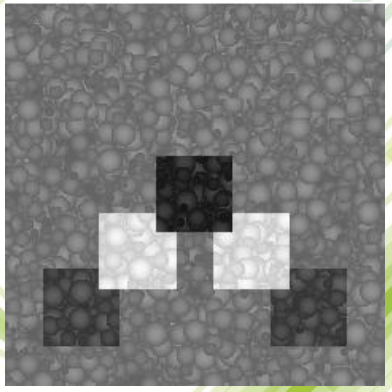
- Questions:
  - Window size (3x3, 5x5, 7x7, … ) ?
  - Search area ?
  - In practice derived experimentally, and additional knowledge.
  - Alternatives:
    - Multiresolution approaches
      (start with large window, then reduce the window size)
    - Pyramidal approaches
      (start with small resolution images, then proceed to higher resolutions, window size is kept constant, however.)

# Stereo matching

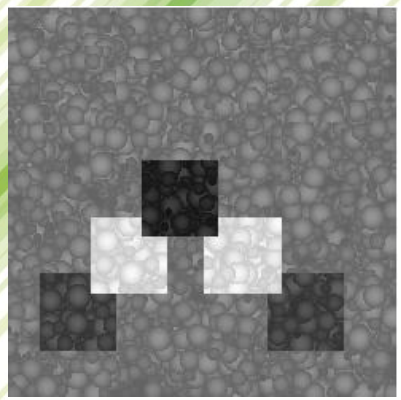
Image pyramid (used in coarse-to-fine approach)



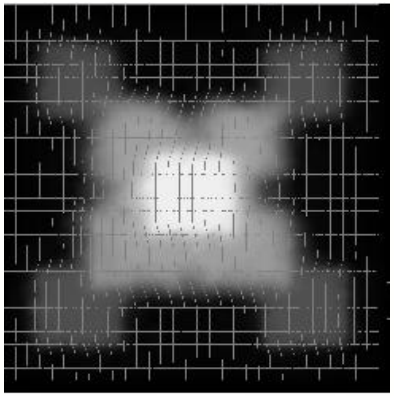
# Stereo matching – an example



Left image



Right image

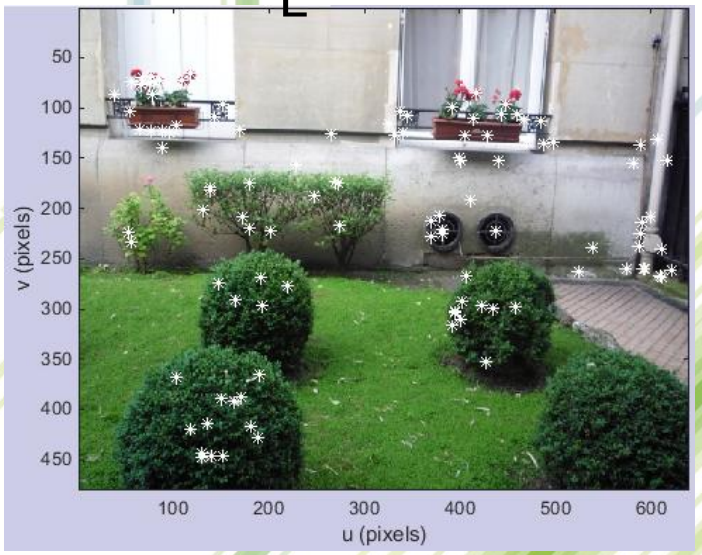


Depth image

# Stereo matching

- Feature based approaches:
  - Detect distinctive image points in both images
  - various ‚detectors', and ‚descriptors' exist
    - edges, lines
    - Corners
  - match these features for each point in one image search for the corresponding point in the other image.
  - Produce sparse depth map,
    - interpolation is needed
  - less false matches
  - less sensitive to photometric variability

# Stereo matching – an example






Key points detected
Epipolar geometry computed
Stereo matching performed
Matched points shown

Produced using P. Corke rvctools for Matlab

# Stereo vision

- Observations:
  - Disparity decreases as distance increases (in case of parallel cameras)
    - Error grows with square of distance (Z)
    - (stereo works well for small distances)
  - As baseline (b) increases, error decreases, but due to perspective, the images are more and more different and consequently, false matches are more likely to occur.

- Some solutions employ ‚multi-baseline' stereo approach.

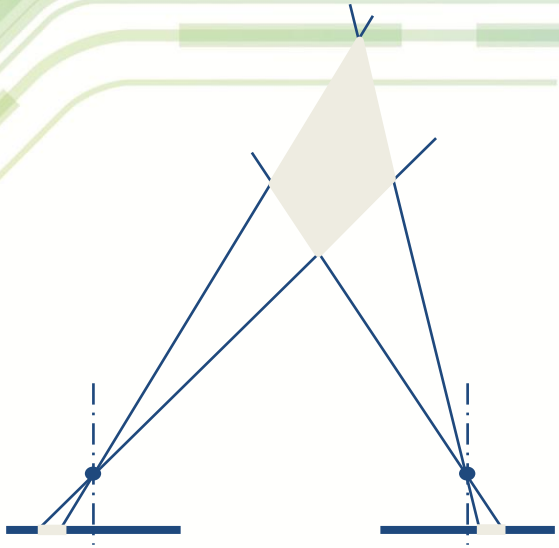
# Stereo vision – analysis of errors

$$Z = \frac{f\,b}{d}$$

$$\Delta Z = \frac{\partial Z}{\partial b}\Delta b + \frac{\partial Z}{\partial f}\Delta f + \frac{\partial Z}{\partial d}\Delta d$$

$$\Delta Z = \left|\frac{f}{d}\right|\Delta b + \left|\frac{b}{d}\right|\Delta f + \left|\frac{f\,b}{d^2}\right|\Delta d$$

$$\Delta Z = \left|\frac{Z}{b}\right|\Delta b + \left|\frac{Z}{f}\right|\Delta f + \left|\frac{Z}{d}\right|\Delta d$$



$$\Delta b, \Delta f \ll \Delta d \qquad (\Delta d = 1\ pixel)\ \text{discretization error}$$

$$\Delta Z \approx \left|\frac{Z}{d}\right|\Delta d = \frac{Z^2}{bf}\Delta d = \frac{Z^2}{bf_u}\lambda_u\,\Delta d$$

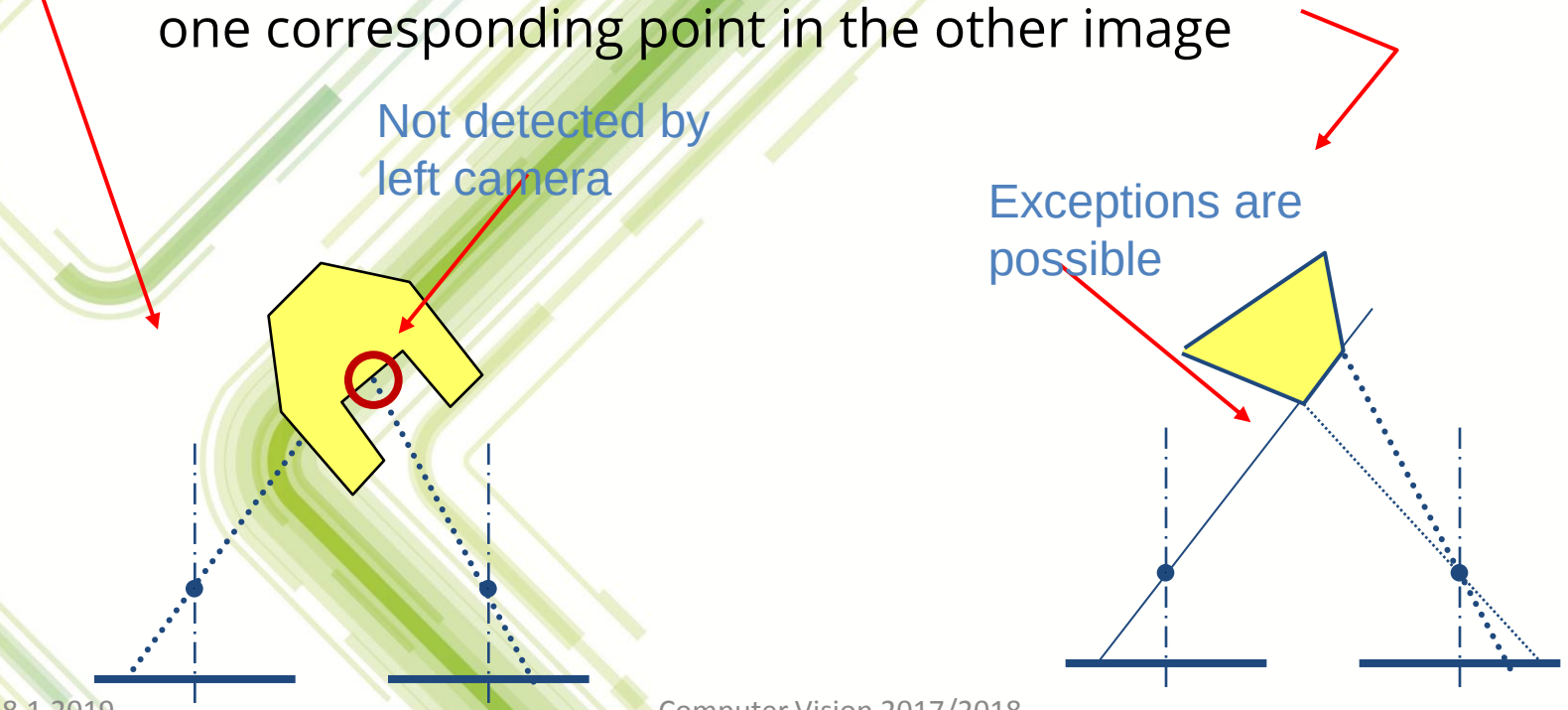
pixel size

focal length in pixels

(discretization error = 1 pixel)

# Key stereo matching problems

- perfect match does not exits,  images are different
- Therefore, additional constraints are helpful
- Epipolar geometry - limits search and false matches
- Uniqueness constraint - point in one image can have at most one corresponding point in the other image



# Questions?