
Mathematical Principles in Visual Computing

Prof. Friedrich Fraundorfer

SS 2024

About me

- Prof. Dr. Friedrich Fraundorfer
- Email: fraundorfer@icg.tugraz.at
- Institute of Computer Graphics and Vision
- Inffeldgasse 16/II
- +43 (316) 873 - **5020**
- Send email to schedule an appointment



Additional lecturers

- Dr. Jörg Müller
- Email: joerg.mueller@icg.tugraz.at
- Institute of Computer Graphics and Vision
- Inffeldgasse 16/II



- Arno Coomans
- Huawei Zürich

Lecture schedule

- 06.03.2024 Fraundorfer
- 13.03.2024 Fraundorfer
- 20.03.2024 Fraundorfer
- 10.04.2024 Fraundorfer
- 17.04.2024 Fraundorfer
- 24.04.2024 Fraundorfer
- 08.05.2024 Müller
- 15.05.2024 Müller
- 22.05.2024 Müller
- 29.05.2024 Müller
- 05.06.2024 Coomans
- 12.06.2024 Coomans
- 19.06.2024 Coomans
- 26.06.2024 Exam

Tutor

- Jun Zhang
- Email: jun.zhang@tugraz.at
- Responsible for questions about classroom assignments
- Q&A slots with tutor
- Q&A in TC forum or e-mail

Course grading

- 3 class room assignments (50% of grade)
 - Math problems
 - Small programming assignments
- Final written exam (50% of grade)
- Written exam at last lecture slot (26 June 2024)
- Submitting the first assignment counts as attempt. A grade will be issued in this case.

Assignments

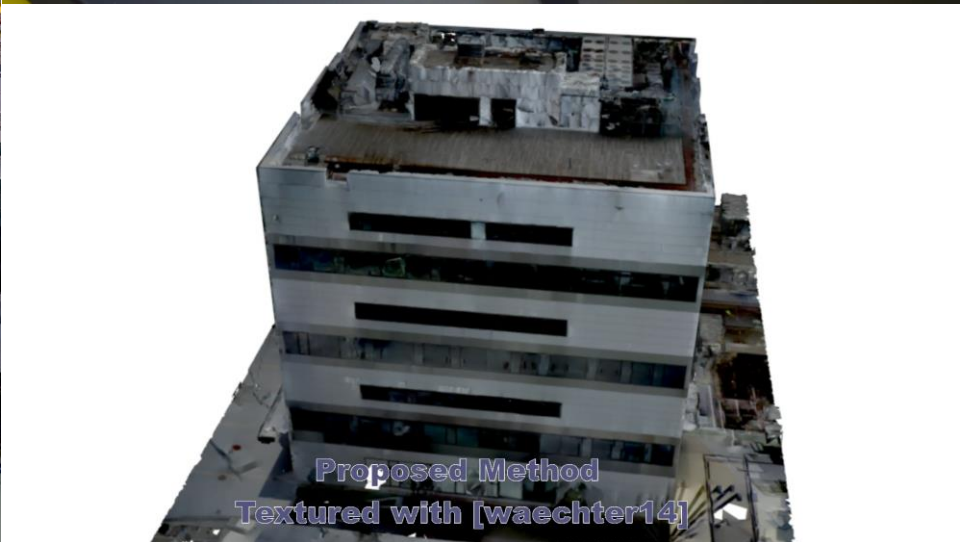
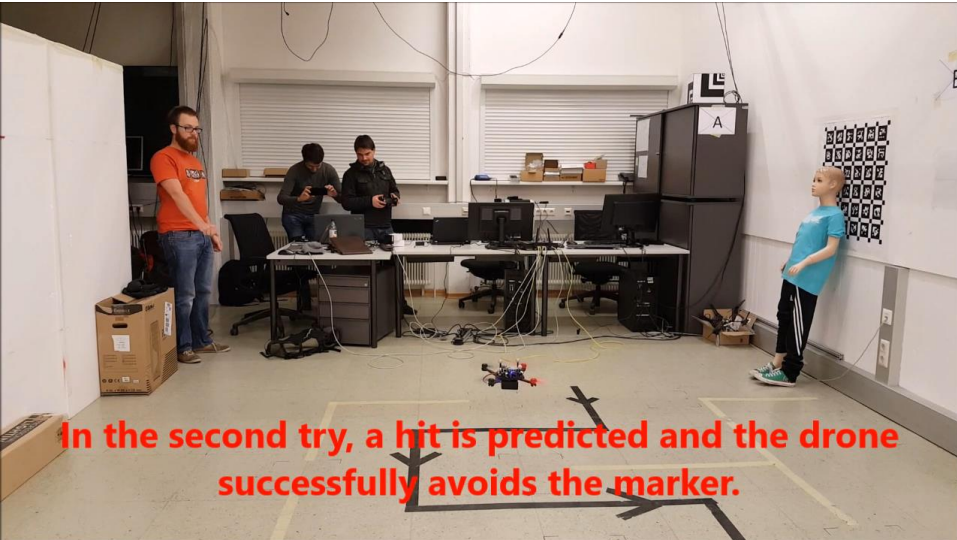
- Individual work, no group work
- Electronic submission using the TeachCenter (Hand-writing and scanning is ok)

- Schedule:
 - Assignment 1
 - Handout: 20.3.2024
 - Deadline: 30.4.2024
 - Assignment 2
 - Handout: 24.4.2024
 - Deadline: 28.5.2024
 - Assignment 3
 - Handout: 22.5.2024
 - Deadline: 18.6.2024

Lecture material

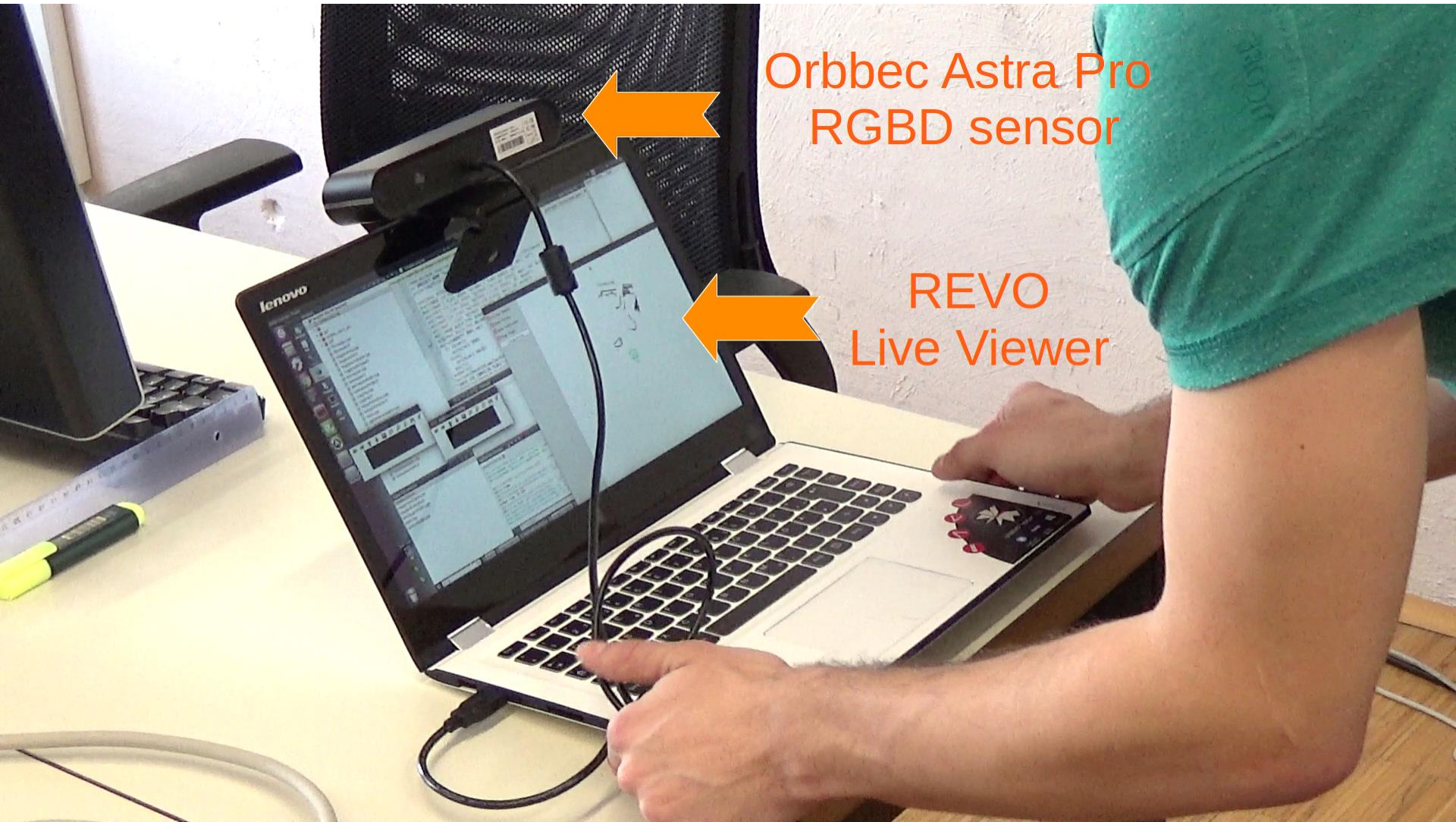
- Slides are the main material
- Links to relevant publications and book sections will be given
- Lecture recordings from last years are available in the Teach Center

Research areas

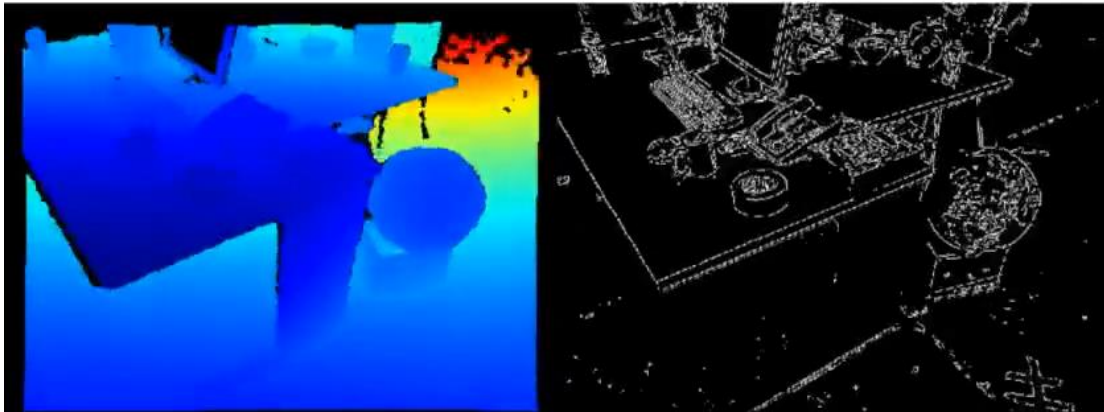
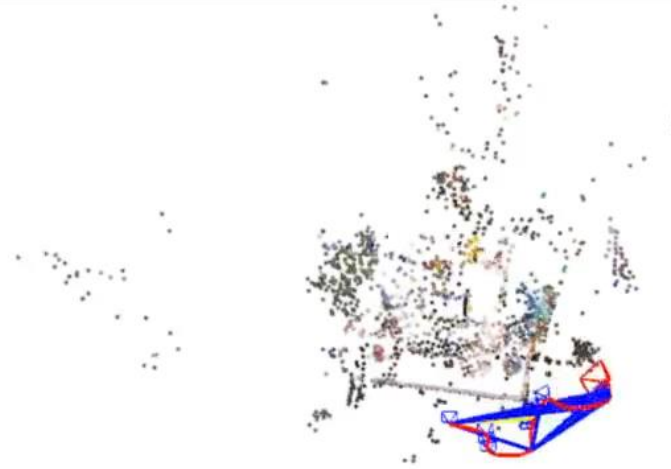


3D scanning - REVO

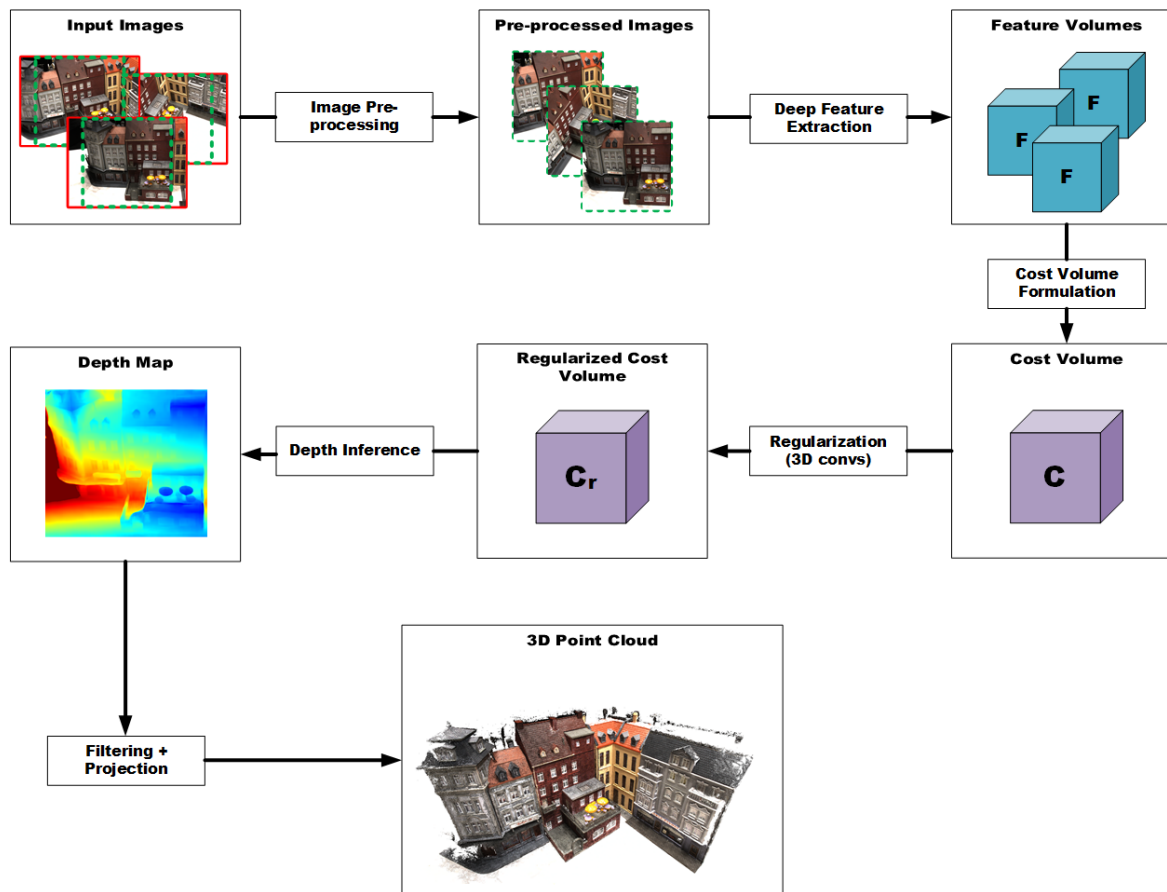
- RGBD recordings with Orbbec Astra Pro



3D scanning - REVO



Multi-View Stereo Pipeline



Bridge inspection

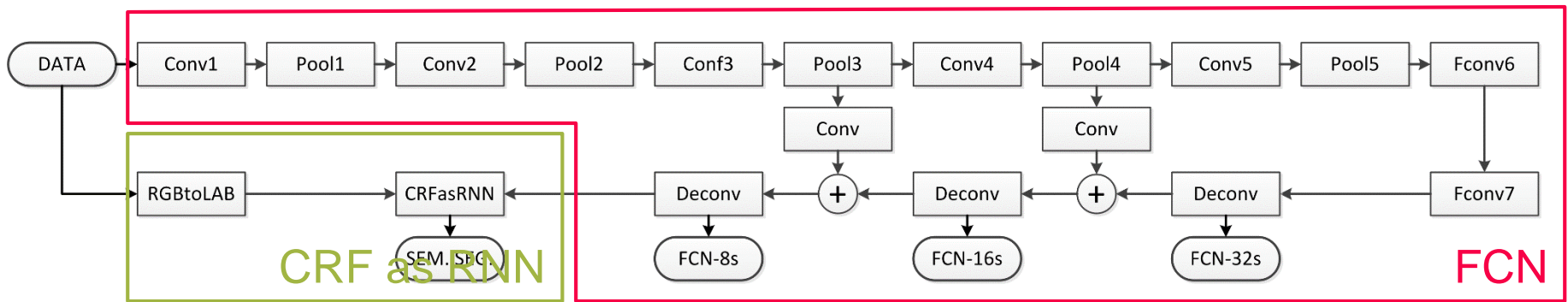
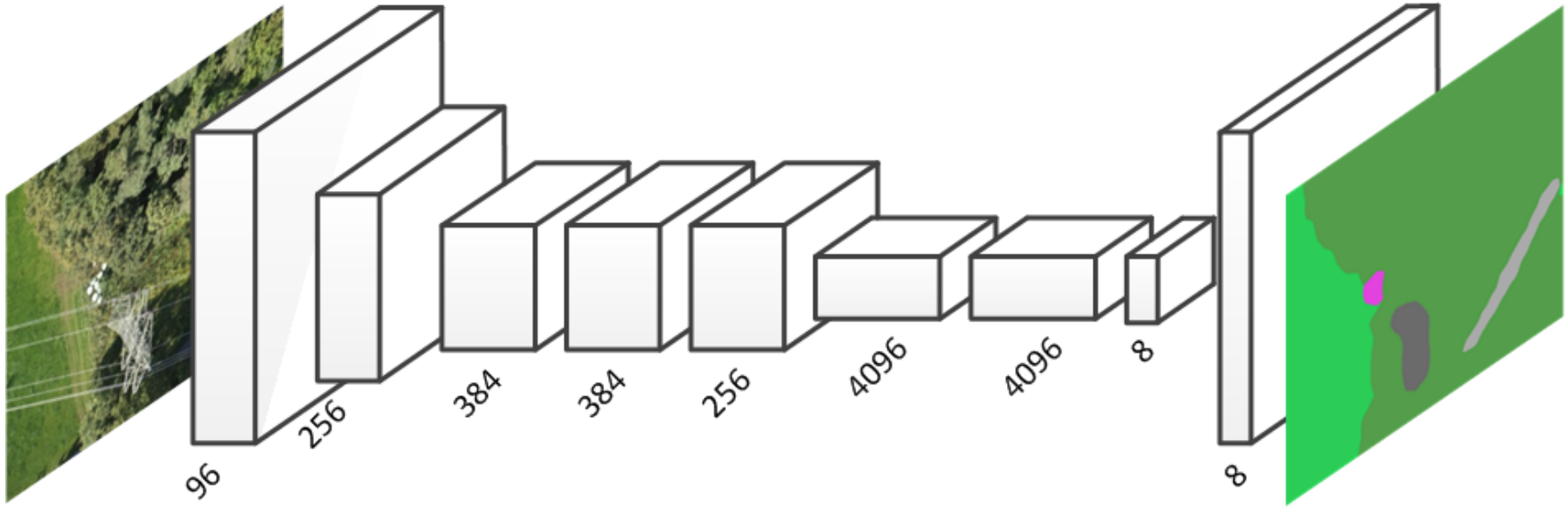


Bridge inspection

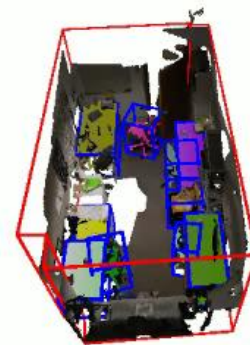
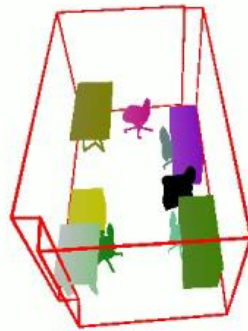




Semantic 3D



3D scene understanding



Embedded AI – Dedicated processors allow integration of deep learning



Embedded AI – Object detection



Topics

- Projective geometry
- Parameterization of rigid transformations
- Polynomial systems in computer vision
- Root-solving
- Projective geometric algebra (Müller)
- Path tracing (Coomans)

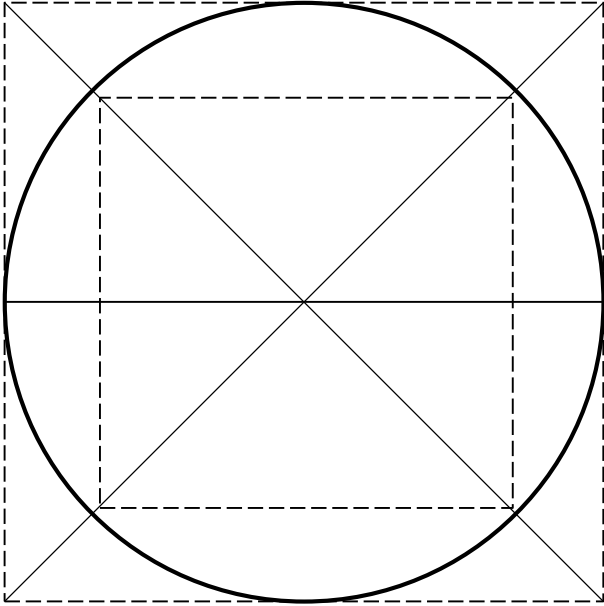
Projective geometry



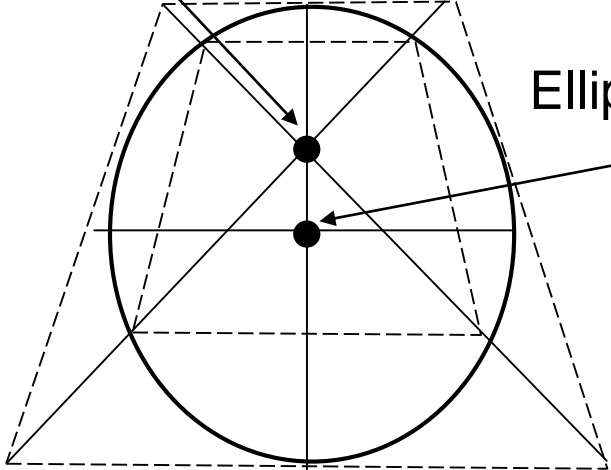
[Source: Flickr]

Projective geometry

Center of projected circle



2D circle

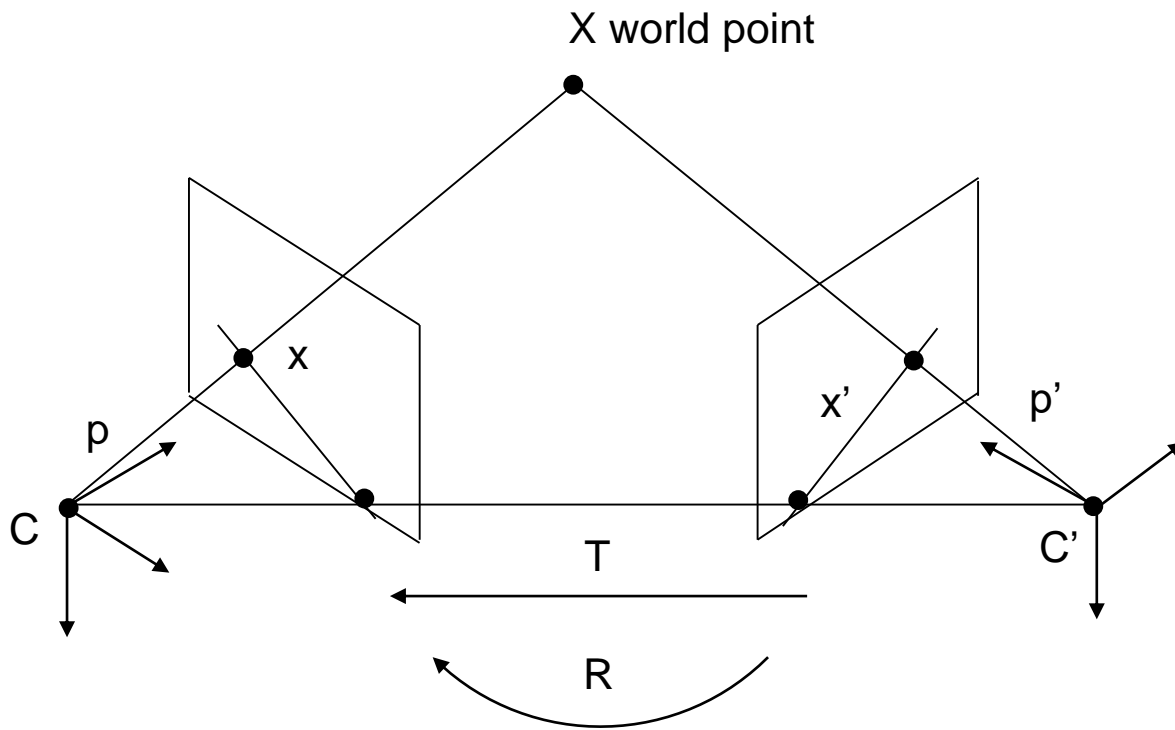


Ellipse center

Circle after projective transformation

Projective geometry

$$x'^T F x = 0 \dots \textit{Epipolar constraint}$$



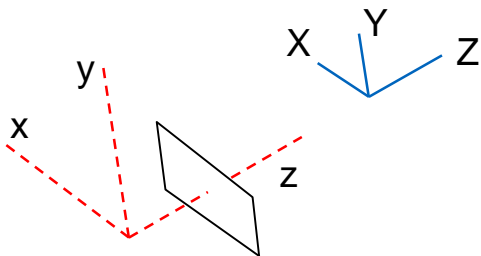
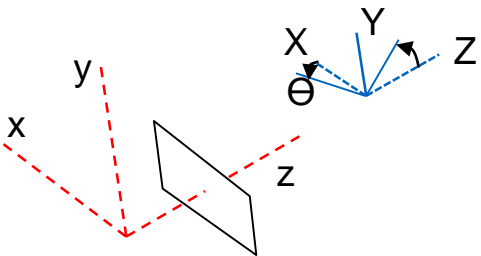
Projective geometry



before rectification

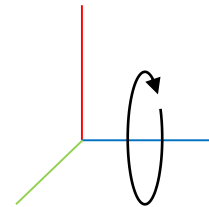


after rectification

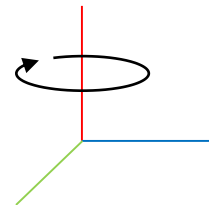


Parameterization of rigid transformations

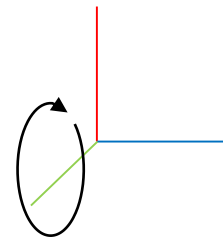
$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix}$$



$$R_y(\beta) = \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix}$$



$$R_z(\gamma) = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

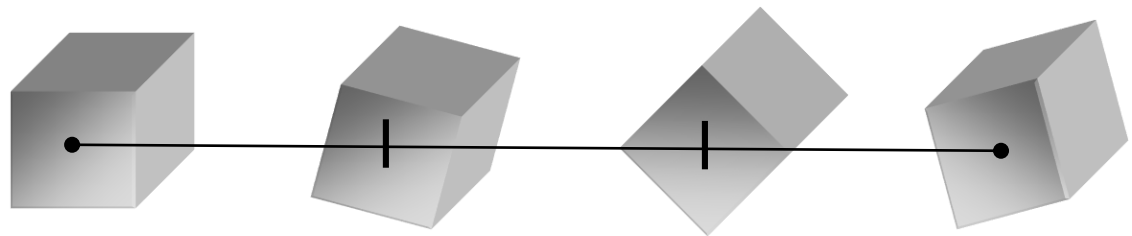


Problems with rotation matrices

- Optimization of rotations (bundle adjustment)

- Newton's method $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

- Linear interpolation

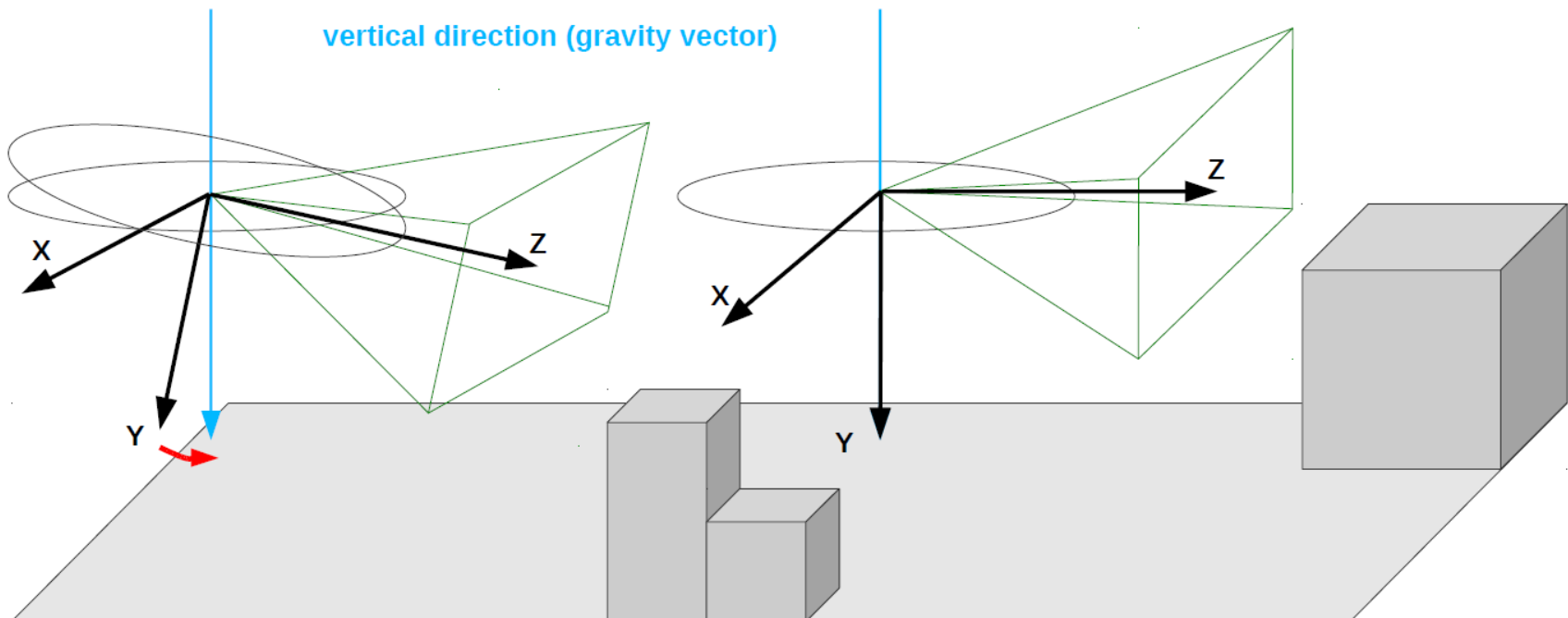


- Filtering and averaging

- E.g. averaging rotation from IMU or camera pose tracker for AR/VR glasses

Polynomial equation systems in computer vision

- Assumption: Ground plane normal to gravity vector, walls are vertical
 - IMU measurements can be used to align camera images/features to gravity vector
 - Motion can be computed from 2pt correspondences on the ground

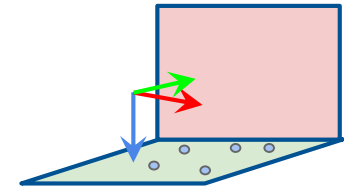


Polynomial equation systems in computer vision

2pt relative pose

$$\boxed{\mathbf{H}} = \boxed{\mathbf{R}_y} \boxed{\mathbf{R}_x \mathbf{R}_z} + \boxed{\mathbf{t}^T} \boxed{\mathbf{n}}$$

Homography Rotation (yaw) Rotation IMU Translation Plane normal



$$\mathbf{H} = \mathbf{R}_y + \mathbf{t}^T \mathbf{n} \quad (\text{for pre-rotated features})$$

$$\boxed{\mathbf{H}} = \boxed{\mathbf{R}_y} + \boxed{[t_x, t_y, t_z]^T} \boxed{[0, 1, 0]} \quad (\mathbf{H} \text{ for ground plane})$$

Homography Rotation (yaw) Translation Plane normal

4 unknowns left, 2 point correspondences give 4 equations

Polynomial systems in computer vision

- P3P, PnP problem:

$$\begin{array}{l} L_1 \\ L_2 \\ L_3 \end{array} \left\{ \begin{array}{l} 2x^2 + y^2 - 2z + 3z^2 + 5 \\ \quad \quad \quad x^2 + z + z^2 \\ \quad \quad \quad x^2y^2 + y^2z^2 - 2 \end{array} \right. = \begin{array}{l} 0 \\ 0 \\ 0 \end{array}$$

- Solution: Reduction to a single polynomial (several schemes)
- Automatic procedure – Gröbner Basis

Polynomial equation systems

Root solving

- 3pt+IMU, 8th degree polynomial
- 6pt generalized camera, 64th degree polynomial

- Fast method: Sturm bracketing

Projective Geometric Algebra

The CoffeeShop

free samples my stash inspect stash

Ganja.js supports operator overloading and algebraic literals.

Operator	Javascript	Name
$a * b$	<code>a*b</code>	Geometric Product
$a \wedge b$	<code>a^b</code>	Outer Product
$a \vee b$	<code>a&b</code>	Regressive Product
$a \cdot b$	<code>a<b</code>	Left Contraction
$a * b * a$	<code>a>>b</code>	Sandwich Product
\tilde{a}	<code>~a</code>	Conjugate
\bar{a}	<code>!a</code>	Dual
\bar{a}	<code>a.Reverse</code>	Reverse
a^{-1}	<code>a**-1</code>	Inverse
e^a	<code>Math.E**a</code>	Exponentiation
$a_{(b)}$	<code>a.Grade(b)</code>	Grade Extraction
$a + b$ or $a - b$	<code>a+b</code> or <code>a-b</code>	Multivector Addition/Subtraction
$4.2e_{12}$	<code>4.2e12</code>	Blade Literals

Ganja.js supports vectors and matrices with multivector elements.

Operator	Javascript	Name
$v = \begin{bmatrix} e_1 & 0 \end{bmatrix}$	<code>v = [1e1,0];</code>	Vector
$A = \begin{bmatrix} 1 & 0 \\ 0 & e_{12} \end{bmatrix}$	<code>A = [[1,0],[0,1e12]];</code>	Matrix
$v \cdot w$	<code>v*w</code>	Vector-Vector dot product.
$A v$	<code>A*v</code>	Matrix-Vector product.
AB	<code>A*B</code>	Matrix-Matrix product.
A^{HT}	<code>~A</code>	Conjugate-Transpose Matrix

Ganja.js can graph 2D and 3D PGA and CGA elements.

```

1- Algebra(3,0,1,())=>{
2   var point = (x,y,z)=>1e123-x*1e012+y*1e013+z*1e023;
3-   var points_to_line = (A, B) => {
4     return A & B;
5   };
6
7   // define three points
8   var A = point(0, 0, -2);
9   var B = point(0, 0, 3);
10  var P = point(0, 2, 0);
11
12  // TODO: implement finding r
13  var l = points_to_line(A, B);
14  var p = l << P;
15  var X = p ^ l;
16  var r = X & P;
17
18  console.log(r);
19
20  var camera = (2e23 + 2e12 + 3e13) / Math.sqrt(17);
21
22-  document.body.appendChild(this.graph(()=>{
23    return [0xA88FF,[A, B],
24            [0x44444,A,"A",B,"B",P,"P",X,"X",
25            [0xFF8888,l,"l",
26            [0x8888FF,p,"p",
27            [0x44AA44,r,"r"
28          ];
29    },{camera}});
30 });

```

Inspektor Konsolle Debugger {} Stilbearbeitung Laufzeitanalyse Speicher Netzwerkanalyse Web-Speicher Barrierefreiheit

Ausgabe filtern

Fehler Warnungen Log Informationen Debug CSS XHR Anfragen

ganja.js line 1437 > eval:15:13

```

Float32Array(16)
0: 0
1: 0
2: 0
3: 0
4: 0
5: 0
6: 0
7: 0
8: 0
9: -50
10: 0
11: 0
12: 0
13: 0
14: 0
15: 0

```