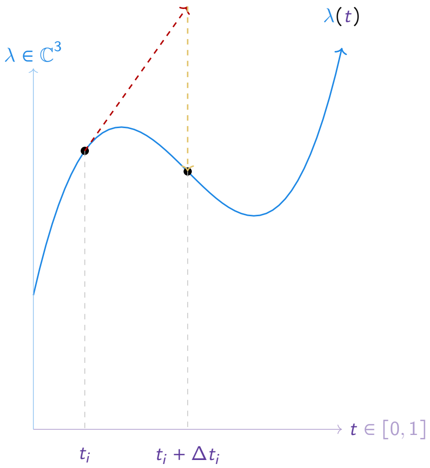
Introduction to homotopy continuation (with a view towards faster solvers)



Tim Duff
U. Washington ⇝ U. Missouri
Monday June 17 2024
CVPR 2024, Seattle

Plan of talk

- Prereqs: multiview geometry at level of Hartley-Ziss. No algebraic geometry!

Plan of talk

- Prereqs: multiview geometry at level of Hartley-Ziss. No algebraic geometry!
- **Part 1 (2:30 – 3:00 PM):** Give an overview of how homotopy continuation works from the user's perspective and some of the general theory, illustrated w/ software demos on familiar examples from 3D vision:
  1. Perspective 3-Point
  2. Five-point essential matrix estimation

Plan of talk

- Prereqs: multiview geometry at level of Hartley-Ziss. No algebraic geometry!
- **Part 1 (2:30 – 3:00 PM):** Give an overview of how homotopy continuation works from the user's perspective and some of the general theory, illustrated w/ software demos on familiar examples from 3D vision:
  1. Perspective 3-Point
  2. Five-point essential matrix estimation
- **Coffee break (3:00 – 4:00 PM):** Come ask us questions!

Plan of talk

- Prereqs: multiview geometry at level of Hartley-Ziss. No algebraic geometry!
- **Part 1 (2:30 – 3:00 PM):** Give an overview of how homotopy continuation works from the user's perspective and some of the general theory, illustrated w/ software demos on familiar examples from 3D vision:
  1. Perspective 3-Point
  2. Five-point essential matrix estimation
- **Coffee break (3:00 – 4:00 PM):** Come ask us questions!
- **Part 2 (4:00 – 4:30 PM):** Showcase some novel applications (and ideas):
  3. 3-view autocalibration w/ partially known intrinsics (minimal relaxation)
  3. Point-line absolute pose (monodromy group)
  5. Radial camera relative pose (minimal solver)

Plan of talk

- Prereqs: multiview geometry at level of Hartley-Ziss. No algebraic geometry!
- **Part 1 (2:30 – 3:00 PM):** Give an overview of how homotopy continuation works from the user's perspective and some of the general theory, illustrated w/ software demos on familiar examples from 3D vision:
    1. Perspective 3-Point
    2. Five-point essential matrix estimation
- **Coffee break (3:00 – 4:00 PM):** Come ask us questions!
- **Part 2 (4:00 – 4:30 PM):** Showcase some novel applications (and ideas):
    3. 3-view autocalibration w/ partially known intrinsics (minimal relaxation)
    3. Point-line absolute pose (monodromy group)
    5. Radial camera relative pose (minimal solver)
- Why should you care about homotopy continuation?

Plan of talk

- Prereqs: multiview geometry at level of Hartley-Ziss. No algebraic geometry!
- **Part 1 (2:30 – 3:00 PM):** Give an overview of how homotopy continuation works from the user's perspective and some of the general theory, illustrated w/ software demos on familiar examples from 3D vision:
  1. Perspective 3-Point
  2. Five-point essential matrix estimation
- **Coffee break (3:00 – 4:00 PM):** Come ask us questions!
- **Part 2 (4:00 – 4:30 PM):** Showcase some novel applications (and ideas):
  3. 3-view autocalibration w/ partially known intrinsics (minimal relaxation)
  3. Point-line absolute pose (monodromy group)
  5. Radial camera relative pose (minimal solver)
- Why should you care about homotopy continuation?
  1. Answer from previous talks: There is a general need for polynomial solvers in 3D vision due to minimal problems being solved in RANSAC. For plenty of problems, homotopy continuation might be the *only* method that works.

Plan of talk

- Prereqs: multiview geometry at level of Hartley-Ziss. No algebraic geometry!
- **Part 1 (2:30 – 3:00 PM):** Give an overview of how homotopy continuation works from the user's perspective and some of the general theory, illustrated w/ software demos on familiar examples from 3D vision:
  1. Perspective 3-Point
  2. Five-point essential matrix estimation
- **Coffee break (3:00 – 4:00 PM):** Come ask us questions!
- **Part 2 (4:00 – 4:30 PM):** Showcase some novel applications (and ideas):
  3. 3-view autocalibration w/ partially known intrinsics (minimal relaxation)
  3. Point-line absolute pose (monodromy group)
  5. Radial camera relative pose (minimal solver)
- Why should you care about homotopy continuation?
  1. Answer from previous talks: There is a general need for polynomial solvers in 3D vision due to minimal problems being solved in RANSAC. For plenty of problems, homotopy continuation might be the *only* method that works.
  2. I would add the following: For plenty of other problems, symbolic computation methods may work better. *Still,* we can uncover useful information from offline HC runs that can be harder to detect symbolically (eg. symmetry.)

My computer vision conference papers:

▲, ● *PLMP* (D., Kohn, Leykin, Pajdla), ICCV 2019.

■, ● *TRPLP* (**Fabbri**, D., **Fan**, ... **Kimia**, ...) CVPR 2020.

▲, ● *PL_1P* (D., Kohn, Leykin, Pajdla) ECCV 2020.

■, ● *Learning to Solve Hard Minimal Problems* (Hruby, D., Leykin, Pajdla) CVPR 2022.

■, ◆, ● *4-view Geom. w/ Unknown Radial Dist.* (Hruby, Korotynskiy, D., ...) CVPR 2023.

▲, ■, ● *Minimal Persp. Autocalibration* (Porfiri dal Cin, D., Magri, Pajdla) **CVPR 2024**.

■, ◆, ● *Efficient solution of point-line absolute pose* (Hruby, D., Pollefeys) **CVPR 2024**.

Legend

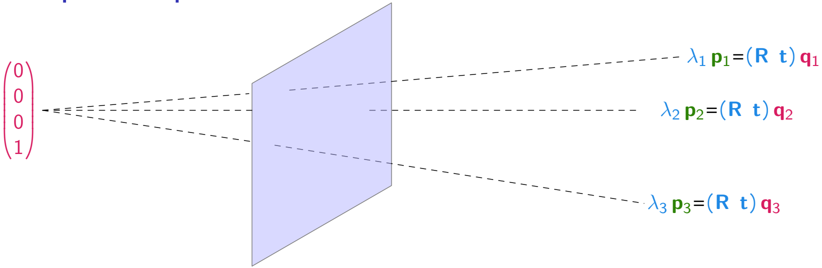
▲: Theory paper: classification of minimal problems.

■: Paper with practical minimal solver(s)

◆: Results relied on Galois/monodromy groups

●: Results relied on numerical homotopy and monodromy computation.

Warm-up: Perspective 3-Point



Given: image points $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3 \in \mathbb{P}^2$, corresponding world points $\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3 \in \mathbb{P}^3$.
(Normalize so that $\mathbf{q}_i = \begin{pmatrix} q_{i1} & q_{i2} & q_{i3} & 1 \end{pmatrix}^T$, and $\mathbf{p}_i{}^T \mathbf{p}_i = 1$.)
Recover: a calibrated camera matrix $(\mathbf{R}\ \ \mathbf{t})$ and scalar depths $\lambda_1, \lambda_2, \lambda_3$ w/

$$\lambda_i \mathbf{p}_i = (\mathbf{R}\ \ \mathbf{t})\mathbf{q}_i.$$

Classical approach of Grunert (1847): eliminate the camera to get 3 equations in 3 unknowns $\lambda = (\lambda_1, \lambda_2, \lambda_3)$, with $8 = 2^3$ solutions: for $1 \le i < j \le 3$,

$$\lambda_i{}^2 + \lambda_j{}^2 - 2\left(\mathbf{p}_i{}^T \mathbf{p}_j\right)\lambda_i \lambda_j = (\mathbf{q}_i - \mathbf{q}_j)^T (\mathbf{q}_i - \mathbf{q}_j).$$

Grunert's equations define a *parametric polynomial system*:

$$\mathbf{f}(\underbrace{\lambda_1, \lambda_2, \lambda_3}_{\text{"variables"},\ \lambda}; \underbrace{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3}_{\text{"parameters"},\ \mathbf{q}, \mathbf{p}}) = \begin{pmatrix} \lambda_1{}^2 + \lambda_2{}^2 - 2\left(\mathbf{p}_1{}^T \mathbf{p}_2\right)\lambda_1\lambda_2 - (\mathbf{q}_1 - \mathbf{q}_2)^T(\mathbf{q}_1 - \mathbf{q}_2) \\ \lambda_1{}^2 + \lambda_3{}^2 - 2\left(\mathbf{p}_1{}^T \mathbf{p}_3\right)\lambda_1\lambda_3 - (\mathbf{q}_1 - \mathbf{q}_3)^T(\mathbf{q}_1 - \mathbf{q}_3) \\ \lambda_2{}^2 + \lambda_3{}^2 - 2\left(\mathbf{p}_2{}^T \mathbf{p}_3\right)\lambda_2\lambda_3 - (\mathbf{q}_2 - \mathbf{q}_3)^T(\mathbf{q}_2 - \mathbf{q}_3) \end{pmatrix}$$

Main ideas behind homotopy continuation:

1. To solve a system for some parameter values of interest $\mathbf{q}^1, \mathbf{p}^1$, it helps if we already know solutions for some other parameters $\mathbf{q}^0, \mathbf{p}^0$.

Grunert's equations define a *parametric polynomial system*:

$$\mathbf{f}(\underbrace{\lambda_1, \lambda_2, \lambda_3}_{\text{``variables'', } \lambda}; \underbrace{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3}_{\text{``parameters'', } \mathbf{q}, \mathbf{p}}) = \begin{pmatrix} \lambda_1{}^2 + \lambda_2{}^2 - 2\left(\mathbf{p}_1{}^T \mathbf{p}_2\right)\lambda_1\lambda_2 - (\mathbf{q}_1 - \mathbf{q}_2)^T(\mathbf{q}_1 - \mathbf{q}_2) \\ \lambda_1{}^2 + \lambda_3{}^2 - 2\left(\mathbf{p}_1{}^T \mathbf{p}_3\right)\lambda_1\lambda_3 - (\mathbf{q}_1 - \mathbf{q}_3)^T(\mathbf{q}_1 - \mathbf{q}_3) \\ \lambda_2{}^2 + \lambda_3{}^2 - 2\left(\mathbf{p}_2{}^T \mathbf{p}_3\right)\lambda_2\lambda_3 - (\mathbf{q}_2 - \mathbf{q}_3)^T(\mathbf{q}_2 - \mathbf{q}_3) \end{pmatrix}$$

Main ideas behind homotopy continuation:

1. To solve a system for some parameter values of interest $\mathbf{q}^1, \mathbf{p}^1$, it helps if we already know solutions for some other parameters $\mathbf{q}^0, \mathbf{p}^0$.
2. Suppose we had a differentiable homotopy function $\mathbf{H}(\lambda; t)$ such that

$$\mathbf{H}(\lambda; 0) = \mathbf{f}(\lambda; \mathbf{q}^0, \mathbf{p}^0), \quad \mathbf{H}(\lambda; 1) = \mathbf{f}(\lambda; \mathbf{q}^1, \mathbf{p}^1).$$

Then we can solve $\mathbf{H}(\lambda; t) = \mathbf{0}$ for λ as an implicit function of t in a neighborhood of any point where the 3×3 Jacobian $\frac{\partial \mathbf{H}}{\partial \lambda}$ is nonsingular :

Grunert's equations define a *parametric polynomial system*:

$$\mathbf{f}(\underbrace{\lambda_1, \lambda_2, \lambda_3}_{\text{"variables"},\ \lambda}; \underbrace{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3}_{\text{"parameters"},\ \mathbf{q},\mathbf{p}}) = \begin{pmatrix} {\lambda_1}^2 + {\lambda_2}^2 - 2\left({\mathbf{p}_1}^T \mathbf{p}_2\right)\lambda_1\lambda_2 - (\mathbf{q}_1 - \mathbf{q}_2)^T(\mathbf{q}_1 - \mathbf{q}_2) \\ {\lambda_1}^2 + {\lambda_3}^2 - 2\left({\mathbf{p}_1}^T \mathbf{p}_3\right)\lambda_1\lambda_3 - (\mathbf{q}_1 - \mathbf{q}_3)^T(\mathbf{q}_1 - \mathbf{q}_3) \\ {\lambda_2}^2 + {\lambda_3}^2 - 2\left({\mathbf{p}_2}^T \mathbf{p}_3\right)\lambda_2\lambda_3 - (\mathbf{q}_2 - \mathbf{q}_3)^T(\mathbf{q}_2 - \mathbf{q}_3) \end{pmatrix}$$

Main ideas behind homotopy continuation:

1. To solve a system for some parameter values of interest $\mathbf{q}^1, \mathbf{p}^1$, it helps if we already know solutions for some other parameters $\mathbf{q}^0, \mathbf{p}^0$.
2. Suppose we had a differentiable homotopy function $\mathbf{H}(\lambda; t)$ such that

$$\mathbf{H}(\lambda; 0) = \mathbf{f}(\lambda; \mathbf{q}^0, \mathbf{p}^0), \quad \mathbf{H}(\lambda; 1) = \mathbf{f}(\lambda; \mathbf{q}^1, \mathbf{p}^1).$$

Then we can solve $\mathbf{H}(\lambda; t) = \mathbf{0}$ for λ as an implicit function of t in a neighborhood of any point where the 3×3 Jacobian $\frac{\partial \mathbf{H}}{\partial \lambda}$ is nonsingular :

$$\mathbf{H} = \mathbf{0} \quad \Rightarrow$$

Grunert's equations define a *parametric polynomial system*:

$$\mathbf{f}(\underbrace{\lambda_1, \lambda_2, \lambda_3}_{\text{“variables”},\ \lambda}; \underbrace{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3}_{\text{“parameters”},\ \mathbf{q},\mathbf{p}}) = \begin{pmatrix} {\lambda_1}^2 + {\lambda_2}^2 - 2\left({\mathbf{p}_1}^T \mathbf{p}_2\right)\lambda_1\lambda_2 - (\mathbf{q}_1 - \mathbf{q}_2)^T(\mathbf{q}_1 - \mathbf{q}_2) \\ {\lambda_1}^2 + {\lambda_3}^2 - 2\left({\mathbf{p}_1}^T \mathbf{p}_3\right)\lambda_1\lambda_3 - (\mathbf{q}_1 - \mathbf{q}_3)^T(\mathbf{q}_1 - \mathbf{q}_3) \\ {\lambda_2}^2 + {\lambda_3}^2 - 2\left({\mathbf{p}_2}^T \mathbf{p}_3\right)\lambda_2\lambda_3 - (\mathbf{q}_2 - \mathbf{q}_3)^T(\mathbf{q}_2 - \mathbf{q}_3) \end{pmatrix}$$

Main ideas behind homotopy continuation:

1. To solve a system for some parameter values of interest $\mathbf{q}^1, \mathbf{p}^1$, it helps if we already know solutions for some other parameters $\mathbf{q}^0, \mathbf{p}^0$.
2. Suppose we had a differentiable homotopy function $\mathbf{H}(\lambda; t)$ such that

$$\mathbf{H}(\lambda; 0) = \mathbf{f}(\lambda; \mathbf{q}^0, \mathbf{p}^0), \quad \mathbf{H}(\lambda; 1) = \mathbf{f}(\lambda; \mathbf{q}^1, \mathbf{p}^1).$$

Then we can solve $\mathbf{H}(\lambda; t) = \mathbf{0}$ for λ as an implicit function of t in a neighborhood of any point where the 3×3 Jacobian $\frac{\partial \mathbf{H}}{\partial \lambda}$ is nonsingular :

$$\mathbf{H} = \mathbf{0} \quad \Rightarrow \quad \frac{\partial}{\partial t}\mathbf{H} = \frac{\partial \mathbf{H}}{\partial \lambda}\frac{\partial \lambda}{\partial t} + \frac{\partial \mathbf{H}}{\partial t} = \mathbf{0}$$

Grunert's equations define a *parametric polynomial system*:

$$\mathbf{f}(\underbrace{\lambda_1, \lambda_2, \lambda_3}_{\text{"variables"},\ \lambda}; \underbrace{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3}_{\text{"parameters"},\ \mathbf{q}, \mathbf{p}}) = \begin{pmatrix} {\lambda_1}^2 + {\lambda_2}^2 - 2\left({\mathbf{p}_1}^T \mathbf{p}_2\right)\lambda_1\lambda_2 - (\mathbf{q}_1 - \mathbf{q}_2)^T(\mathbf{q}_1 - \mathbf{q}_2) \\ {\lambda_1}^2 + {\lambda_3}^2 - 2\left({\mathbf{p}_1}^T \mathbf{p}_3\right)\lambda_1\lambda_3 - (\mathbf{q}_1 - \mathbf{q}_3)^T(\mathbf{q}_1 - \mathbf{q}_3) \\ {\lambda_2}^2 + {\lambda_3}^2 - 2\left({\mathbf{p}_2}^T \mathbf{p}_3\right)\lambda_2\lambda_3 - (\mathbf{q}_2 - \mathbf{q}_3)^T(\mathbf{q}_2 - \mathbf{q}_3) \end{pmatrix}$$

Main ideas behind homotopy continuation:

1. To solve a system for some parameter values of interest $\mathbf{q}^1, \mathbf{p}^1$, it helps if we already know solutions for some other parameters $\mathbf{q}^0, \mathbf{p}^0$.
2. Suppose we had a differentiable homotopy function $\mathbf{H}(\lambda; t)$ such that

$$\mathbf{H}(\lambda; 0) = \mathbf{f}(\lambda; \mathbf{q}^0, \mathbf{p}^0), \quad \mathbf{H}(\lambda; 1) = \mathbf{f}(\lambda; \mathbf{q}^1, \mathbf{p}^1).$$

Then we can solve $\mathbf{H}(\lambda; t) = \mathbf{0}$ for λ as an implicit function of t in a neighborhood of any point where the 3×3 Jacobian $\frac{\partial \mathbf{H}}{\partial \lambda}$ is nonsingular :

$$\mathbf{H} = \mathbf{0} \quad \Rightarrow \quad \frac{\partial}{\partial t}\mathbf{H} = \frac{\partial \mathbf{H}}{\partial \lambda}\frac{\partial \lambda}{\partial t} + \frac{\partial \mathbf{H}}{\partial t} = \mathbf{0}$$

$$\Rightarrow \lambda = -\int \left(\left(\frac{\partial \mathbf{H}}{\partial \lambda} \right)^{-1} \frac{\partial \mathbf{H}}{\partial t} \right) dt$$

Main algorithm Parameter homotopy for a square system $\mathbf{f}$ with parameters $\mathbf{q}, \mathbf{p}$:

Main algorithm Parameter homotopy for a square system $\mathbf{f}$ with parameters $\mathbf{q}, \mathbf{p}$:

Step 1. (offline) Solve the problem for *start parameters* $\mathbf{q}^0, \mathbf{p}^0$ (random, complex-valued.)

Main algorithm Parameter homotopy for a square system $\mathbf{f}$ with parameters $\mathbf{q}, \mathbf{p}$:

Step 1. (offline) Solve the problem for *start parameters* $\mathbf{q}^0, \mathbf{p}^0$ (random, complex-valued.)

$\rightsquigarrow$ Starting problem-solution pairs $(\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,1}), \ldots, (\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,8})$.

Main algorithm Parameter homotopy for a square system $\mathbf{f}$ with parameters $\mathbf{q}, \mathbf{p}$:

Step 1. (offline) Solve the problem for *start parameters* $\mathbf{q}^0, \mathbf{p}^0$ (random, complex-valued.)

$\rightsquigarrow$ Starting problem-solution pairs $(\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,1}), \ldots, (\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,8})$.

Step 2. (online) Given new, real-valued parameters $\mathbf{q}^1, \mathbf{p}^1$, define a parameter homotopy

$$\mathbf{H}(\lambda; t) = f\left(\lambda; (1-t)\mathbf{q}^0 + t\mathbf{q}^1, (1-t)\mathbf{p}^0 + t\mathbf{p}^1\right)$$

Main algorithm Parameter homotopy for a square system $\mathbf{f}$ with parameters $\mathbf{q}, \mathbf{p}$:

Step 1. (offline) Solve the problem for *start parameters* $\mathbf{q}^0, \mathbf{p}^0$ (random, complex-valued.)

$\rightsquigarrow$ Starting problem-solution pairs $(\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,1}), \ldots, (\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,8})$.

Step 2. (online) Given new, real-valued parameters $\mathbf{q}^1, \mathbf{p}^1$, define a parameter homotopy

$$\mathbf{H}(\lambda; t) = f\left(\lambda; (1-t)\mathbf{q}^0 + t\mathbf{q}^1, (1-t)\mathbf{p}^0 + t\mathbf{p}^1\right)$$

This interpolates (non-linearly) between: $\mathbf{H}|_{t=0} = \mathbf{f}(\lambda; \mathbf{q}^0, \mathbf{p}^0)$ and $\mathbf{H}|_{t=1} = \mathbf{f}(\lambda; \mathbf{q}^1, \mathbf{p}^1)$.

Main algorithm Parameter homotopy for a square system $\mathbf{f}$ with parameters $\mathbf{q}, \mathbf{p}$:

Step 1. (offline) Solve the problem for *start parameters* $\mathbf{q}^0, \mathbf{p}^0$ (random, complex-valued.)

$\rightsquigarrow$ Starting problem-solution pairs $(\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,1}), \ldots, (\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,8})$.

Step 2. (online) Given new, real-valued parameters $\mathbf{q}^1, \mathbf{p}^1$, define a parameter homotopy

$$\mathbf{H}(\lambda; t) = f\left(\lambda; (1-t)\mathbf{q}^0 + t\mathbf{q}^1, (1-t)\mathbf{p}^0 + t\mathbf{p}^1\right)$$

This interpolates (non-linearly) between: $\mathbf{H}|_{t=0} = \mathbf{f}(\lambda; \mathbf{q}^0, \mathbf{p}^0)$ and $\mathbf{H}|_{t=1} = \mathbf{f}(\lambda; \mathbf{q}^1, \mathbf{p}^1)$.

Now, numerically integrate the ODE system

$$\lambda'(t) = -\left(\frac{\partial \mathbf{H}}{\partial \lambda}\right)^{-1} \frac{\partial \mathbf{H}}{\partial t}$$

with initial conditions $\lambda(0) \in \{\lambda^{0,1}, \ldots, \lambda^{0,8}\}$

Main algorithm Parameter homotopy for a square system $\mathbf{f}$ with parameters $\mathbf{q}, \mathbf{p}$:

Step 1. (offline) Solve the problem for *start parameters* $\mathbf{q}^0, \mathbf{p}^0$ (random, complex-valued.)
$\rightsquigarrow$ Starting problem-solution pairs $(\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,1}), \ldots, (\mathbf{q}^0, \mathbf{p}^0, \lambda^{0,8})$.

Step 2. (online) Given new, real-valued parameters $\mathbf{q}^1, \mathbf{p}^1$, define a parameter homotopy

$$\mathbf{H}(\lambda; t) = f\left(\lambda; (1-t)\mathbf{q}^0 + t\mathbf{q}^1, (1-t)\mathbf{p}^0 + t\mathbf{p}^1\right)$$

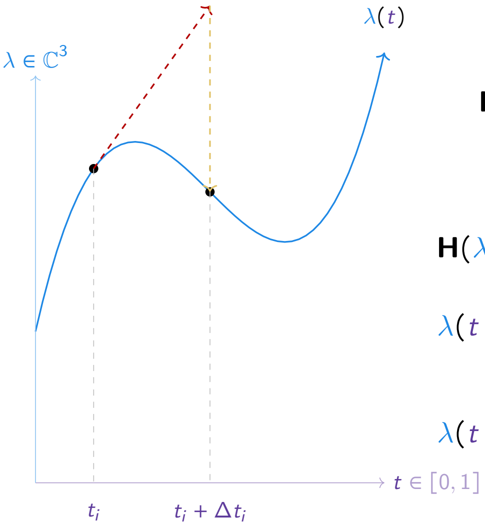
This interpolates (non-linearly) between: $\mathbf{H}|_{t=0} = \mathbf{f}(\lambda; \mathbf{q}^0, \mathbf{p}^0)$ and $\mathbf{H}|_{t=1} = \mathbf{f}(\lambda; \mathbf{q}^1, \mathbf{p}^1)$.

Now, numerically integrate the ODE system

$$\lambda'(t) = -\left(\frac{\partial \mathbf{H}}{\partial \lambda}\right)^{-1} \frac{\partial \mathbf{H}}{\partial t}$$

with initial conditions $\lambda(0) \in \{\lambda^{0,1}, \ldots, \lambda^{0,8}\}$
$\rightsquigarrow$ Target problem-solution pairs $(\mathbf{q}^0, \mathbf{p}^0, \lambda^{1,1}), \ldots, (\mathbf{q}^0, \mathbf{p}^0, \lambda^{1,8})$.

"Path-tracking"— Numerical Integration via Predictor/Corrector



$$\mathbf{H}(\lambda; t) = f\left(\lambda; (1-t)\mathbf{q}^0 + t\mathbf{q}^1, (1-t)\mathbf{p}^0 + t\mathbf{p}^1\right)$$

$$\lambda'(t) = -\left(\frac{\partial \mathbf{H}}{\partial \lambda}\right)^{-1} \frac{\partial \mathbf{H}}{\partial t} \qquad \text{(ODE)}$$

$$\mathbf{H}(\lambda(0); 0) = 0 \qquad \text{(IC)}$$

$$\lambda(t + \Delta t_i) \leftarrow \lambda(t_i) - \Delta t_i \left(\left(\frac{\partial \mathbf{H}}{\partial \lambda}\right)^{-1} \frac{\partial \mathbf{H}}{\partial t}\right)\Bigg|_{t=t_i+\Delta t_i} \qquad \text{(predict)}$$

$$\lambda(t + \Delta t_i) \leftarrow \lambda(t + \Delta t_i) - \left(\left(\frac{\partial \mathbf{H}}{\partial \lambda}\right)^{-1} \mathbf{H}\right)\Bigg|_{t=t_i+\Delta t_i} \qquad \text{(correct)}$$

Starting parameters and solutions can be found using *monodromy* as follows:

Starting parameters and solutions can be found using *monodromy* as follows:

1. Synthesize a random scene $\mathbf{q}$ and camera $(\mathbf{R} \;\; \mathbf{t})$. Compute associated λ and $\mathbf{p}$.

Starting parameters and solutions can be found using *monodromy* as follows:

1. Synthesize a random scene $\mathbf{q}$ and camera $(\mathbf{R} \;\; \mathbf{t})$. Compute associated λ and $\mathbf{p}$.
2. Starting from the problem-solution pair $(\mathbf{q}, \mathbf{p}, \lambda)$, track paths back and forth between other sets of random parameter values. Eventually, you will pick up all solutions. This is because the space of problem-solution pairs is *connected*.

Starting parameters and solutions can be found using *monodromy* as follows:

1. Synthesize a random scene $\mathbf{q}$ and camera $(\mathbf{R} \ \mathbf{t})$. Compute associated λ and $\mathbf{p}$.
2. Starting from the problem-solution pair $(\mathbf{q}, \mathbf{p}, \lambda)$, track paths back and forth between other sets of random parameter values. Eventually, you will pick up all solutions. This is because the space of problem-solution pairs is *connected*.

Simplest example: monodromy of $x^2 = p$.

Two important points:

1. **The initial monodromy solve only needs to be done once.**
2. Monodromy permutes solutions, and **structure preserved by these permutations can be exploited for solving.** Taking P3P as an example, the monodromy permutations preserve a non-trivial partition of solutions, namely

$$\{\lambda^1, \lambda^2, \lambda^3, \lambda^4\} \cup \{-\lambda^1, -\lambda^2, -\lambda^3, -\lambda^4\}$$

   This means we can track 4 paths instead of 8.

Parameter homotopy: Remarks and Caveats

- Tweak-able subroutines: adaptive stepsize, predictor (Runge-Kutta, Padé, ...), ...

Parameter homotopy: Remarks and Caveats

- Tweak-able subroutines: adaptive stepsize, predictor (Runge-Kutta, Padé, ...), ...
- Pretty much everything still works for systems of *rational functions*.

Parameter homotopy: Remarks and Caveats

- Tweak-able subroutines: adaptive stepsize, predictor (Runge-Kutta, Padé, ...), ...
- Pretty much everything still works for systems of *rational functions*.
- *Assumes* input data ("parameters" $\mathbf{q}, \mathbf{p}$) are sufficiently generic—for data that are special or degenerate, the Jacobian could become singular!

Parameter homotopy: Remarks and Caveats

- Tweak-able subroutines: adaptive stepsize, predictor (Runge-Kutta, Padé, ...), ...
- Pretty much everything still works for systems of *rational functions*.
- *Assumes* input data ("parameters" $\mathbf{q}, \mathbf{p}$) are sufficiently generic—for data that are special or degenerate, the Jacobian could become singular!
- In theory, the method can find all isolated solutions to a polynomial system over the complex numbers. In practice, solutions may get lost due to inherent limitations of the numerical methods used and floating-point arithmetic.

Parameter homotopy: Remarks and Caveats

- Tweak-able subroutines: adaptive stepsize, predictor (Runge-Kutta, Padé, ...), ...
- Pretty much everything still works for systems of *rational functions*.
- *Assumes* input data ("parameters" $\mathbf{q}, \mathbf{p}$) are sufficiently generic—for data that are special or degenerate, the Jacobian could become singular!
- In theory, the method can find all isolated solutions to a polynomial system over the complex numbers. In practice, solutions may get lost due to inherent limitations of the numerical methods used and floating-point arithmetic.

Parameter homotopy: Remarks and Caveats

- Tweak-able subroutines: adaptive stepsize, predictor (Runge-Kutta, Padé, ...), ...
- Pretty much everything still works for systems of *rational functions*.
- *Assumes* input data ("parameters" $\mathbf{q}, \mathbf{p}$) are sufficiently generic—for data that are special or degenerate, the Jacobian could become singular!
- In theory, the method can find all isolated solutions to a polynomial system over the complex numbers. In practice, solutions may get lost due to inherent limitations of the numerical methods used and floating-point arithmetic.

Implementations:

- General-purpose software packages:
  - `Bertini`
  - `HomotopyContinuation` for Julia language
  - **Macaulay2 (computer algebra system, covered in this talk)**
    → **Some relevant packages:** `NumericalAlgebraicGeometry`, `MonodromySolver`.
  - `PHCPack`

Parameter homotopy: Remarks and Caveats

- Tweak-able subroutines: adaptive stepsize, predictor (Runge-Kutta, Padé, ...), ...
- Pretty much everything still works for systems of *rational functions*.
- *Assumes* input data ("parameters" $\mathbf{q}, \mathbf{p}$) are sufficiently generic—for data that are special or degenerate, the Jacobian could become singular!
- In theory, the method can find all isolated solutions to a polynomial system over the complex numbers. In practice, solutions may get lost due to inherent limitations of the numerical methods used and floating-point arithmetic.

Implementations:

- General-purpose software packages:
  - `Bertini`
  - `HomotopyContinuation` for Julia language
  - **Macaulay2 (computer algbera system, covered in this talk)**
    → **Some relevant packages:** `NumericalAlgebraicGeometry`, `MonodromySolver`.
  - `PHCPack`
- None of the above are suitable for RANSAC! More specialized libraries:
  - **MiNuS (Ricardo's talk)**, plus various derivatives
  - **GPU-HC (Hongyi's talk)**

Five-point essential matrix estimation

$$D_1(\mathbf{E}) = \ldots = D_9(\mathbf{E}) = \mathbf{p}_{11}{}^T\mathbf{E}\mathbf{p}_{21} = \ldots = \mathbf{p}_{15}{}^T\mathbf{E}\mathbf{p}_{25} = \ell(\mathbf{E}) - 1 = 0, \qquad (1)$$

where $\mathbf{E}$ is 3×3, ℓ is a linear form, and $D_1, \ldots D_9$ are the Demazure constraints,

$$2\mathbf{E}\mathbf{E}^T\mathbf{E} - \operatorname{tr}\left(\mathbf{E}\mathbf{E}^T\right)\mathbf{E} = 0. \qquad (2)$$

Five-point essential matrix estimation

$$D_1(\mathbf{E}) = \ldots = D_9(\mathbf{E}) = \mathbf{p}_{11}{}^T\mathbf{E}\mathbf{p}_{21} = \ldots = \mathbf{p}_{15}{}^T\mathbf{E}\mathbf{p}_{25} = \ell(\mathbf{E}) - 1 = 0, \tag{1}$$

where $\mathbf{E}$ is 3×3, ℓ is a linear form, and $D_1, \ldots D_9$ are the Demazure constraints,

$$2\mathbf{E}\mathbf{E}^T\mathbf{E} - \operatorname{tr}\left(\mathbf{E}\mathbf{E}^T\right)\mathbf{E} = 0. \tag{2}$$

Q: 15 equations in 9 unknowns: Is that a problem?

Five-point essential matrix estimation

$$D_1(\mathbf{E}) = \ldots = D_9(\mathbf{E}) = \mathbf{p}_{11}{}^T \mathbf{E} \mathbf{p}_{21} = \ldots = \mathbf{p}_{15}{}^T \mathbf{E} \mathbf{p}_{25} = \ell(\mathbf{E}) - 1 = 0, \qquad (1)$$

where $\mathbf{E}$ is 3×3, ℓ is a linear form, and $D_1, \ldots D_9$ are the Demazure constraints,

$$2\mathbf{E}\mathbf{E}^T\mathbf{E} - \operatorname{tr}\left(\mathbf{E}\mathbf{E}^T\right)\mathbf{E} = 0. \qquad (2)$$

Q: 15 equations in 9 unknowns: Is that a problem?

A: No! If $(\mathbf{p}^*, \mathbf{E}^*)$ is a random problem-solution pair, we have

$$\operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \,\middle|\, \frac{\partial \mathbf{f}}{\partial \mathbf{E}}\right)\Bigg|_{(\mathbf{p},\mathbf{E})=(\mathbf{p}^*,\mathbf{E}^*)} = \operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{E}}\right)\Bigg|_{(\mathbf{p},\mathbf{E})=(\mathbf{p}^*,\mathbf{E}^*)} = 9 = \#\text{variables}.$$

Five-point essential matrix estimation

$$D_1(\mathbf{E}) = \ldots = D_9(\mathbf{E}) = \mathbf{p}_{11}{}^T \mathbf{E} \mathbf{p}_{21} = \ldots = \mathbf{p}_{15}{}^T \mathbf{E} \mathbf{p}_{25} = \ell(\mathbf{E}) - 1 = 0, \tag{1}$$

where $\mathbf{E}$ is 3×3, ℓ is a linear form, and $D_1, \ldots D_9$ are the Demazure constraints,

$$2\mathbf{E}\mathbf{E}^T\mathbf{E} - \operatorname{tr}\left(\mathbf{E}\mathbf{E}^T\right)\mathbf{E} = 0. \tag{2}$$

Q: 15 equations in 9 unknowns: Is that a problem?
A: No! If $(\mathbf{p}^*, \mathbf{E}^*)$ is a random problem-solution pair, we have

$$\operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \,\middle|\, \frac{\partial \mathbf{f}}{\partial \mathbf{E}}\right)\Bigg|_{(\mathbf{p},\mathbf{E})=(\mathbf{p}^*,\mathbf{E}^*)} = \operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{E}}\right)\Bigg|_{(\mathbf{p},\mathbf{E})=(\mathbf{p}^*,\mathbf{E}^*)} = 9 = \#\text{variables}.$$

These *rank equations* can be used to check that a problem is well-posed ("minimal") and that we can use a parameter homotopy based on a full-rank square subsystem of $\mathbf{f}$.

Five-point essential matrix estimation

$$D_1(\mathbf{E}) = \ldots = D_9(\mathbf{E}) = \mathbf{p}_{11}{}^T \mathbf{E}\mathbf{p}_{21} = \ldots = \mathbf{p}_{15}{}^T \mathbf{E}\mathbf{p}_{25} = \ell(\mathbf{E}) - 1 = 0, \tag{1}$$

where $\mathbf{E}$ is 3×3, ℓ is a linear form, and $D_1, \ldots D_9$ are the Demazure constraints,

$$2\mathbf{E}\mathbf{E}^T\mathbf{E} - \operatorname{tr}\left(\mathbf{E}\mathbf{E}^T\right)\mathbf{E} = 0. \tag{2}$$

Q: 15 equations in 9 unknowns: Is that a problem?
A: No! If $(\mathbf{p}^*, \mathbf{E}^*)$ is a random problem-solution pair, we have

$$\operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \;\middle|\; \frac{\partial \mathbf{f}}{\partial \mathbf{E}}\right)\Bigg|_{(\mathbf{p},\mathbf{E})=(\mathbf{p}^*,\mathbf{E}^*)} = \operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{E}}\right)\Bigg|_{(\mathbf{p},\mathbf{E})=(\mathbf{p}^*,\mathbf{E}^*)} = 9 = \#\text{variables}.$$

These *rank equations* can be used to check that a problem is well-posed ("minimal") and that we can use a parameter homotopy based on a full-rank square subsystem of $\mathbf{f}$. **Moreover, the 10 solutions to the subsystem which satisfy the original system are not connected by monodromy permutations to any excess solutions.**

Software Demo and/or Break Time

- Coffee break (3:00 – 4:00 PM). Come ask us questions!
- Software tutorial in *Macaulay2*: 5-point problem with HC. No installation required! Will run in browser!
- Part 2 (4:00 – 4:30 PM): Novel applications
  3. 3-view autocalibration w/ partially known intrinsics
  4. Point-line absolute pose
  5. Radial camera relative pose

3-view Autocalibration

Given: five point triplets in three views, $\mathbf{p}_{i,1} \leftrightarrow \mathbf{p}_{i,2} \leftrightarrow \mathbf{p}_{i,3}$, $1 \leq i \leq 5$, w/

$$\lambda_{i,j}\mathbf{p}_{i,j} = \mathbf{K}\begin{pmatrix}\mathbf{R}_j & \mathbf{t}_j\end{pmatrix}\mathbf{q}_i, \qquad \textbf{recover } \mathbf{K} = \begin{pmatrix} f & 0 & u \\ 0 & f & v \\ 0 & 0 & 1 \end{pmatrix}.$$

3-view Autocalibration

Given: five point triplets in three views, $\mathbf{p}_{i,1} \leftrightarrow \mathbf{p}_{i,2} \leftrightarrow \mathbf{p}_{i,3}$, $1 \le i \le 5$, w/

$$\lambda_{i,j}\mathbf{p}_{i,j} = \mathbf{K}\begin{pmatrix}\mathbf{R}_j & \mathbf{t}_j\end{pmatrix}\mathbf{q}_i, \qquad \textbf{recover } \mathbf{K} = \begin{pmatrix} f & 0 & u \\ 0 & f & v \\ 0 & 0 & 1 \end{pmatrix}.$$

(Porfiri dal Cin, D., et al, CVPR '24): for $1 \le i_1 < i_2 \le 5$, $2 \le j \le 3$,

$$\left\|\mathbf{K}^{-1}\left(\lambda_{i_1,1}\mathbf{p}_{i_1,1} - \lambda_{i_2,1}\mathbf{p}_{i_2,1}\right)\right\|^2 - \left\|\mathbf{K}^{-1}\left(\lambda_{i_1,j}\mathbf{p}_{i_1,j} - \lambda_{i_2,j}\mathbf{p}_{i_2,j}\right)\right\|^2 = 0.$$

3-view Autocalibration

Given: five point triplets in three views, $\mathbf{p}_{i,1} \leftrightarrow \mathbf{p}_{i,2} \leftrightarrow \mathbf{p}_{i,3}$, $1 \le i \le 5$, w/

$$\lambda_{i,j}\mathbf{p}_{i,j} = \mathbf{K}\begin{pmatrix}\mathbf{R}_j & \mathbf{t}_j\end{pmatrix}\mathbf{q}_i, \qquad \textbf{recover } \mathbf{K} = \begin{pmatrix} f & 0 & u \\ 0 & f & v \\ 0 & 0 & 1 \end{pmatrix}.$$

(Porfiri dal Cin, D., et al, CVPR '24): for $1 \le i_1 < i_2 \le 5$, $2 \le j \le 3$,

$$\left\|\mathbf{K}^{-1}\left(\lambda_{i_1,1}\mathbf{p}_{i_1,1} - \lambda_{i_2,1}\mathbf{p}_{i_2,1}\right)\right\|^2 - \left\|\mathbf{K}^{-1}\left(\lambda_{i_1,j}\mathbf{p}_{i_1,j} - \lambda_{i_2,j}\mathbf{p}_{i_2,j}\right)\right\|^2 = 0.$$

Q: 21 equations in 18 unknowns — is this a problem?

3-view Autocalibration

Given: five point triplets in three views, $\mathbf{p}_{i,1} \leftrightarrow \mathbf{p}_{i,2} \leftrightarrow \mathbf{p}_{i,3}$, $1 \leq i \leq 5$, w/

$$\lambda_{i,j}\mathbf{p}_{i,j} = \mathbf{K}\begin{pmatrix}\mathbf{R}_j & \mathbf{t}_j\end{pmatrix}\mathbf{q}_i, \qquad \textbf{recover } \mathbf{K} = \begin{pmatrix} f & 0 & u \\ 0 & f & v \\ 0 & 0 & 1 \end{pmatrix}.$$

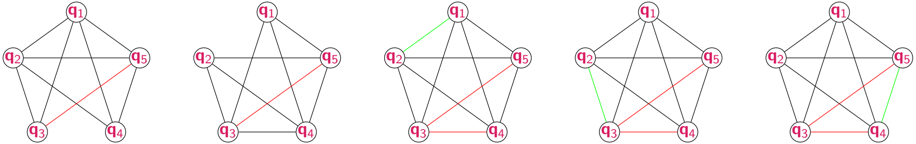
(Porfiri dal Cin, D., et al, CVPR '24): for $1 \leq i_1 < i_2 \leq 5$, $2 \leq j \leq 3$,

$$\left\|\mathbf{K}^{-1}\left(\lambda_{i_1,1}\mathbf{p}_{i_1,1} - \lambda_{i_2,1}\mathbf{p}_{i_2,1}\right)\right\|^2 - \left\|\mathbf{K}^{-1}\left(\lambda_{i_1,j}\mathbf{p}_{i_1,j} - \lambda_{i_2,j}\mathbf{p}_{i_2,j}\right)\right\|^2 = 0.$$

Q: 21 equations in 18 unknowns — is this a problem?
A: This time, yes! A rank condition fails because the problem is overconstrained:

$$\operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial(\mathbf{K},\lambda)}\right) = 18 < 19 = \operatorname{rank}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \,\middle|\, \frac{\partial \mathbf{f}}{\partial(\mathbf{K},\lambda)}\right)$$

Autocalibration (cont.)



Black edge—enforce distance constraint for view-pairs $(1, 2)$ and $(1, 3)$
Red edge—enforce distance constraint for view-pair $(1, 2)$ only
Green edge—enforce distance constraint for view-pair $(1, 3)$ only

Unlike the case of a well-constrained problem (such as five-point relative pose), for a square relaxation of deleted equations need not be enforced.
The number of solutions for each relaxation can vary significantly:

`ffuv0`: min = 16118, max = 119119.

	Fountain-P11							Herz-Jesu-P8						
Method	Δfg	Δuv	Δs	Re_{gt}	Re	ϵ_R	ϵ_C	Δfg	Δuv	Δs	Re_{gt}	Re	ϵ_R	ϵ_C
Kruppa-6	0.137	0.184	0.022	19.563	2.891	7.061	5.579	0.098	0.112	0.014	14.565	1.112	2.125	1.902
Kruppa-7	0.249	0.204	0.040	28.197	-	-	-	0.122	0.114	0.040	15.252	-	-	-
Kruppa-8	0.260	0.173	0.029	28.466	-	-	-	0.140	0.115	0.022	13.606	-	-	-
Kruppa BnB	0.127	0.058	0.014	9.231	-	-	-	0.078	0.096	0.018	21.023	-	-	-
Modulus BnB	0.162	0.071	0.016	10.540	-	-	-	0.097	0.102	0.019	22.641	-	-	-
`ffuv0`	**0.017**	**0.029**	-	**4.435**	**0.449**	**0.623**	**0.664**	**0.017**	**0.044**	-	**8.082**	**0.672**	**0.664**	**0.656**
`fguv0`	0.028	0.050	-	8.580	0.554	0.970	1.183	0.029	0.063	-	11.128	0.680	1.295	1.540
`fguvs`	0.035	0.064	**0.008**	9.769	1.075	1.274	1.428	0.041	0.058	**0.013**	11.348	0.989	1.085	1.139

Absolute Pose with points and lines

Given p 3D-2D point correspondences, and l 3D-2D line correspondences, recover the calibrated camera that produced them. We get a minimal problem when $p + l = 3$. Monodromy permutations were computed for all four minimal problems,
(D., Korotynskiy, Pajdla, Regan, SIAM J. Appl. Alg. Geom., 2023)

Problem	p	l	# solutions/$\mathbb{C}$	monodromy group	hidden symmetry?
P3P	3	0	8	$\mathbf{S_2 \wr S_4 \cap A_8}$	yes
P2P1L	2	1	4	$\mathbf{S_2 \wr S_2}$	**yes**
P1P2L	1	2	8	$\mathbf{S_2 \wr S_4 \cap A_8}$	**yes**
P3L	0	3	8	$\mathbf{S_8}$	**no**

Absolute Pose with Points and Lines (cont.)

Prior work (Ramalingam et al., ICRA ’11) proposed degree-4 / degree-8 solvers for P2P1L / P1P2L. *Although we do not theoretically prove that our solutions are of the lowest possible degrees, we believe...*

Can symmetries detected by monodromy lead to a practical symbolic solver?

Absolute Pose with Points and Lines (cont.)

Prior work (Ramalingam et al., ICRA '11) proposed degree-4 / degree-8 solvers for P2P1L / P1P2L. *Although we do not theoretically prove that our solutions are of the lowest possible degrees, we believe...*


Can symmetries detected by monodromy lead to a practical symbolic solver?
Yes—(Hruby, D., Pollefeys, CVPR 2024).

Method	Avg.	Min	Max
P2P1L Ours	**314**	**231**	**3061**
P2P1L Poselib	1861	1439	10102
P2P1L Ramalingam	8898	5805	49984
P1P2L Ours	**504**	**364**	**4554**
P1P2L Poselib	1967	1484	12931

Table: Solver timings in nanoseconds

Method	Avg. $\mathbf{R}_{err}$	Avg. $\mathbf{t}_{rel}$
P2P1L Ours	**5.3e-12**	**3.7e-10**
P2P1L Poselib	2.8e-05	2.0e-05
P2P1L Ramalingam	4.7e-07	2.3e-05
PP1P2L Ours	**1.2e-07**	**2.0e-06**
P1P2L Poselib	3.3e-05	3.4e-05

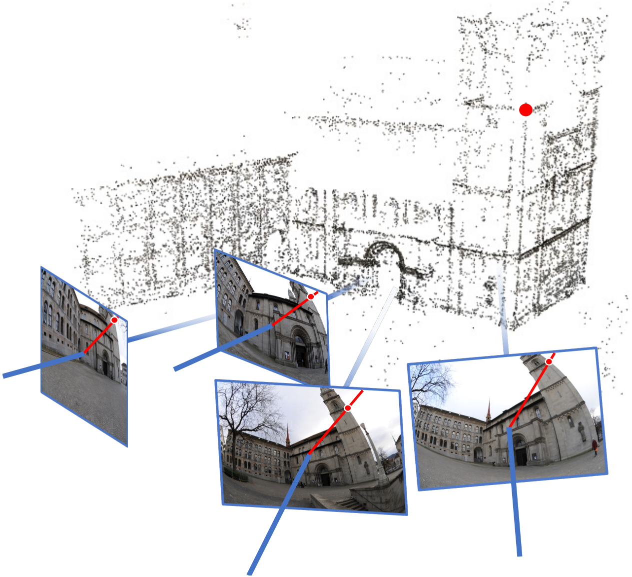
Table: Average solver errors ($\mathbf{R}_{err}$ in radians.)

That was nice...

But what about problems where homotopy continuation is *needed as a solver*?

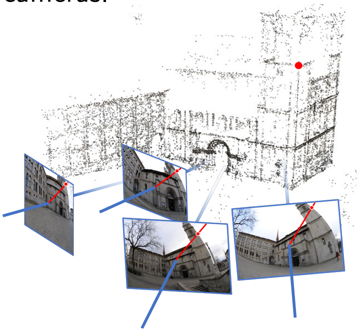
That was nice...

But what about problems where homotopy continuation is *needed as a solver*?



(Hruby, Korotynskiy, D., Oeding, Pollefeys, Pajdla, Larsson, CVPR 2023) The minimal *relative pose* problem for calibrated radial cameras has **3584** (complex) solutions, but can be solved by tracking just **28** paths.

Projective geometry for radially-distorted, *non-pinhole* cameras.



(Pollefeys, Thirthala, '09) A *radial camera* is a surjective, projective linear map $\mathbf{A} : \mathbb{P}^3 \dashrightarrow \mathbb{P}^1$.

Assume image distortion is *radially symmetric* about some known origin—WLOG $[0 : 0 : 1] \in \mathbb{P}^2$.
Distorted points move along radial lines.

$$\operatorname{span}\left\{ \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{pmatrix} \mathbf{q}, \quad \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right\} =_{\mathbb{P}^2}$$

$$\operatorname{span}\left\{ \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{0}_3 \end{pmatrix} \mathbf{q}, \quad \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right\}$$

$$\therefore \qquad \mathbf{q} \mapsto \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \end{pmatrix} \mathbf{q}$$

A pinhole camera determines a radial camera—simply drop the last row of the camera matrix!
In other words, we may relax the pinhole model by considering only constraints on radial lines.

Consider four pinhole projections of a common 3D point,

$$\mathbf{A}_i \mathbf{q} = \lambda_i \mathbf{p}_i, \quad i = 1, \ldots, 4.$$

Let $\mathcal{R}(\mathbf{A}_1), \ldots, \mathcal{R}(\mathbf{A}_4)$ denote the associated radial cameras.

$$\begin{pmatrix} \mathcal{R}(\mathbf{A}_1) & \mathbf{p}_1 & & & \\ \mathcal{R}(\mathbf{A}_2) & & \mathbf{p}_2 & & \\ \mathcal{R}(\mathbf{A}_3) & & & \mathbf{p}_3 & \\ \mathcal{R}(\mathbf{A}_4) & & & & \mathbf{p}_4 \end{pmatrix} \begin{pmatrix} \mathbf{q} \\ -\lambda_1 \\ -\lambda_2 \\ -\lambda_3 \\ -\lambda_4 \end{pmatrix} = \mathbf{0}_{8\times 1}$$

$$\Rightarrow \quad \det \begin{pmatrix} \mathcal{R}(\mathbf{A}_1) & \mathbf{p}_1 & & & \\ \mathcal{R}(\mathbf{A}_2) & & \mathbf{p}_2 & & \\ \mathcal{R}(\mathbf{A}_3) & & & \mathbf{p}_3 & \\ \mathcal{R}(\mathbf{A}_4) & & & & \mathbf{p}_4 \end{pmatrix} = 0.$$

This is a quadrilinear form in the $\mathbb{P}^1$-image coordinates, represented by the $2 \times 2 \times 2 \times 2$ *radial quadrifocal tensor*. These are very special tensors with special internal constraints: they parametrize a 13-dimensional space $Y \subset \mathbb{P}^{2\times2\times2\times2-1} = \mathbb{P}^{15}$. They also must satisfy complicated internal constraints—see (Lin-Sturmfels, J. Alg. '09).

For **either calibrated or uncalibrated radial cameras**, 13 matches are minimal:

$$13 = 4 \cdot (4 \cdot 2 - 1) - (4 \cdot 4 - 1) = 4 \cdot (3 + 2) - 7.$$

For **either calibrated or uncalibrated radial cameras**, 13 matches are minimal:

$$13 = 4 \cdot (4 \cdot 2 - 1) - (4 \cdot 4 - 1) = 4 \cdot (3 + 2) - 7.$$

Parametric polynomial system

$$\det \begin{pmatrix} \mathcal{R}(\mathbf{A}_1) & \mathbf{p}_{i,1} & & & \\ \mathcal{R}(\mathbf{A}_2) & & \mathbf{p}_{i,2} & & \\ \mathcal{R}(\mathbf{A}_3) & & & \mathbf{p}_{i,3} & \\ \mathcal{R}(\mathbf{A}_4) & & & & \mathbf{p}_{i,4} \end{pmatrix} = 0, \quad i = 1, \ldots, 13.$$

Number of solutions?

1. $3584 = 2^7 \cdot 28$ in calibrated cameras
2. $56 = 2 \cdot 28$ in uncalibrated cameras
3. 28 in quadrifocal tensors

Parameter homotopies give us "the best of both worlds": we can use the simple equations above describing (1)–(2), but only need to track 28 paths as in (3).

Final Thoughts

Homotopy continuation isn't currently the method of choice for solving (minimal) geometric estimation problems in vision. But there are reasons to be optimistic:

Final Thoughts

Homotopy continuation isn't currently the method of choice for solving (minimal) geometric estimation problems in vision. But there are reasons to be optimistic:

1. Scales better to problems with more solutions than symbolic methods.

Final Thoughts

Homotopy continuation isn't currently the method of choice for solving (minimal) geometric estimation problems in vision. But there are reasons to be optimistic:

1. Scales better to problems with more solutions than symbolic methods.
2. Not all applications require RANSAC runtimes (example: autocalibration.)

Final Thoughts

Homotopy continuation isn't currently the method of choice for solving (minimal) geometric estimation problems in vision. But there are reasons to be optimistic:

1. Scales better to problems with more solutions than symbolic methods.
2. Not all applications require RANSAC runtimes (example: autocalibration.)
3. Still can be useful for other reasons:
   - Designing other solvers (example: point-line absolute pose.)
   - "Fallback" methods for traditional SfM **(Fabbri, D., Fan, et al., CVPR 2020)**.
   - Measuring algebraic "hardness" of problems.

Final Thoughts

Homotopy continuation isn't currently the method of choice for solving (minimal) geometric estimation problems in vision. But there are reasons to be optimistic:

1. Scales better to problems with more solutions than symbolic methods.
2. Not all applications require RANSAC runtimes (example: autocalibration.)
3. Still can be useful for other reasons:
   - Designing other solvers (example: point-line absolute pose.)
   - "Fallback" methods for traditional SfM **(Fabbri, D., Fan, et al., CVPR 2020)**.
   - Measuring algebraic "hardness" of problems.
4. Current and future work improving overall efficiency:
   4.1 Learning starting problem-solution pairs **(Hruby, D., et al., CVPR 2022)**.
   4.2 Parallelization and GPU
      - **(Chien, Fan et al., CVPR 2022.)**
      - **(Ding, Chien, et al., ICCV 2023.)**