A Direct Least-Squares Solution to Multi-View Absolute and Relative Pose from 2D-3D Perspective Line Pairs

Research Group on Visual Computation

Hichem Abdellali, Robert Frohlich, Zoltan Kato

Institute of Informatics, University of Szeged, Hungary

Problem Statement

❖ **Absolute** and **relative pose** estimation of a **multi-view perspective camera system** using 3D-2D line-pairs

❖ We propose a direct least squares solution **(call it MRPnL)** which:

- ❖ **uses Grobner** basis
- ❖ works for any **n ≥ 3** number of **lines**
- ❖ is suitable for hypothesis testing like in **RANSAC**

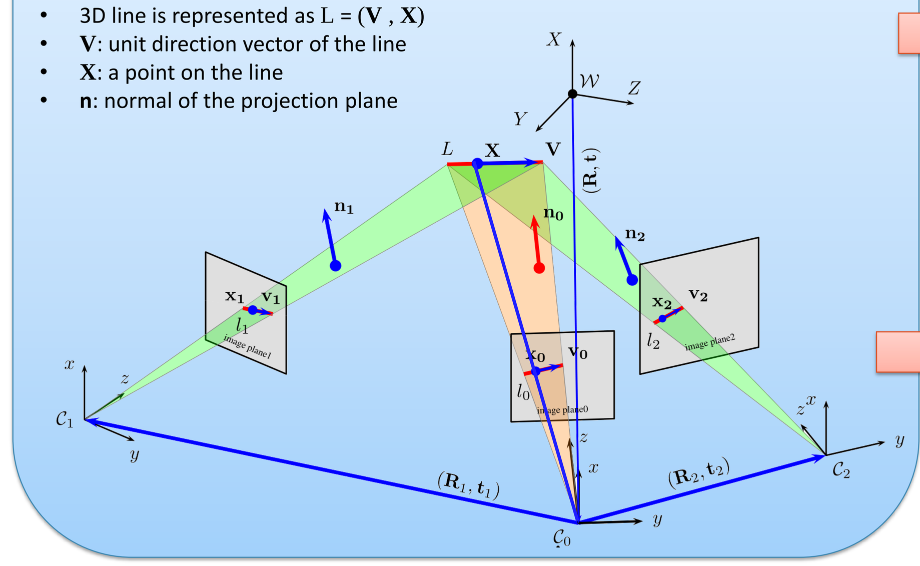
❖ The poses can be further **refined** through a few iterations of an **iterative least squares solver (MRPnL LM)**

Projection of Lines and Camera Pose

• **Idea: two** equations based on **two** different geometric observations

Assumption:
Calibration is known ➔ we work with the normalized image coordinates: $\mathbf{x} = \mathbf{K}^{-1}\mathbf{P}\mathbf{X} = [\mathbf{R}|\mathbf{t}]\mathbf{X}$

- 3D line is represented as $L = (\mathbf{V}, \mathbf{X})$
- $\mathbf{V}$: unit direction vector of the line
- $\mathbf{X}$: a point on the line
- $\mathbf{n}$: normal of the projection plane



1. The **direction vector** of the 3D line and **the normal vector** of the projection plane is **perpendicular**

$$\mathbf{n}^\top \mathbf{R}\mathbf{V} = 0,$$

➔ *only rotation*

2. The vector from the camera center C_0 to an arbitrary point X on line L is also lying on the plane, thus it is also perpendicular to **the normal of the plane**

$$\mathbf{n}^\top(\mathbf{R}\mathbf{X} + \mathbf{t}) = 0$$

➔ *rotation, translation*

- Overall we will have 2 equations for the reference camera :

$$\mathbf{n}^\top \mathbf{R}\mathbf{V} = 0, \qquad \mathbf{n}^\top(\mathbf{R}\mathbf{X} + \mathbf{t}) = 0$$

In the same way we expand it for the other cameras *i*

$$\mathbf{n}_i^\top \mathbf{R}_i \mathbf{R}\mathbf{V} = 0, \qquad \mathbf{n}_i^\top(\mathbf{R}_i(\mathbf{R}\mathbf{X} + \mathbf{t}) + \mathbf{t}_i) = 0$$

Normalization

- 2D image data is normalized by definition as we work on the normalized image plane
- all the 3D line data is transformed into a unit cube, by a translation to the origin and uniform scaling factor

$$s = \frac{1}{max(|h|,|w|,|d|)}.$$

- the result pose $(\tilde{\mathbf{R}}, \tilde{\mathbf{t}})$ has to be denormalized

Robust Outlier Filtering

- we used the error measure proposed in [1]
- it calculates the mean of the shortest distances $d_{\mathbf{x}_s}$ and $d_{\mathbf{x}_e}$ from the 2D line segment endpoints $\mathbf{x}_s$ and $\mathbf{x}_e$ to the corresponding infinite line determined by the backprojected 3D line onto the normalized plane

$$\frac{d_{\mathbf{x}_s} + d_{\mathbf{x}_e}}{2(\|\mathbf{x}_e - \mathbf{x}_s\|)}$$

- the error is normalized with the length of the 2D line segment

Proposed Solution: MRPnL

Direct Least Squares Solver

We reduce the number of unknowns in $\mathbf{R}$ to 2 by defining an intermediate coordinate system $\mathcal{M}$ and precalculating $\mathbf{R}_\mathcal{M}$ using the longest projected line

1 The origin of $\mathcal{M}$ is located at:

$$\mathbf{Y}_\mathcal{M} = \frac{\mathbf{n}_0^\mathcal{C}}{\|\mathbf{n}_0^\mathcal{C}\|}, \quad \mathbf{X}_\mathcal{M} = \frac{\mathbf{n}_0^\mathcal{C} \times \mathbf{V}_0^\mathcal{W}}{\|\mathbf{n}_0^\mathcal{C} \times \mathbf{V}_0^\mathcal{W}\|}, \quad \mathbf{Z}_\mathcal{M} = \frac{\mathbf{X}_\mathcal{M} \times \mathbf{Y}_\mathcal{M}}{\|\mathbf{X}_\mathcal{M} \times \mathbf{Y}_\mathcal{M}\|}$$

$\mathbf{R}_\mathcal{M}$ rotates the normals and direction vectors into the intermediate frame $\mathcal{M}$

$$\mathbf{R}_\mathcal{M} = [\mathbf{X}_\mathcal{M}, \mathbf{Y}_\mathcal{M}, \mathbf{Z}_\mathcal{M}]^\top$$

The rotation $\mathbf{R}_x^\mathcal{M}$ around $\mathbf{X}$ axis within $\mathcal{M}$ is then easily calculated because it is the angle between the $\mathbf{Z}$ axis and $\mathbf{V}_0^\mathcal{M}$

2 the rotation matrix acting within the intermediate coordinate frame $\mathcal{M}$ is composed

$$(1+s^2)(1+r^2)\mathbf{R}^\mathcal{M} = \mathbf{R}_y^\mathcal{M}(s)\mathbf{R}_z^\mathcal{M}(r) = \begin{bmatrix} (1-s^2)(1-r^2) & -2r(1-s^2) & 2s(r^2+1) \\ 2r(s^2+1) & (s^2+1)(1-r^2) & 0 \\ -2s(1-r^2) & 4sr & (1-s^2)(r^2+1) \end{bmatrix}$$

$$(\mathbf{R}_\mathcal{M}\mathbf{n})^\top \mathbf{R}^\mathcal{M}(\mathbf{R}_x^\mathcal{M}\mathbf{R}_\mathcal{M}\mathbf{V}) = \mathbf{n}^{\mathcal{M}\top}\mathbf{R}^\mathcal{M}\mathbf{V}^\mathcal{M} = 0$$

3 Expanding equation in (2) gives a 4th order polynomial of (s, r) with coefficients in terms of $\mathbf{n}^\mathcal{M}$ and $\mathbf{V}^\mathcal{M}$

$$\mathbf{a}^\top\mathbf{u} = \begin{bmatrix} V_1 n_1 + V_2 n_2 + V_3 n_3 \\ 2V_1 n_2 - 2V_2 n_1 \\ 2V_3 n_1 - 2V_1 n_3 \\ -V_1 n_1 - V_2 n_2 + V_3 n_3 \\ -V_1 n_1 + V_2 n_2 - V_3 n_3 \\ V_1 n_1 - V_2 n_2 - V_3 n_3 \\ 2V_1 n_2 + 2V_2 n_1 \\ 2V_3 n_1 + 2V_1 n_3 \\ 4V_2 n_3 \end{bmatrix}^\top \cdot \begin{bmatrix} 1 \\ r \\ s \\ r^2 \\ s^2 \\ s^2r^2 \\ s^2r \\ sr^2 \\ sr \end{bmatrix} = 0$$

Each line-pair generates one such equation

4 This yield to a system of N equations, which is solved in the least squares sense by taking the sum of squares of the system

$$e(s,r) = \sum_{i=1}^{N}(\mathbf{a}_i^\top\mathbf{u})^2 \quad \Rightarrow \quad \nabla e(s,r) = \begin{bmatrix} \sum_{i=1}^{N} \mathbf{b}_{\mathbf{s}_i}^\top \mathbf{u}_\mathbf{s} \\ \sum_{i=1}^{N} \mathbf{b}_{\mathbf{r}_i}^\top \mathbf{u}_\mathbf{r} \end{bmatrix} = \mathbf{0}$$

The solution of the system of the two 7th order polynomial equations provides the rotation parameters (s, r)

5 Once the solution(s) are obtained, the complete $\mathbf{R}$, acting between the world $\mathcal{W}$ and camera frame $\mathcal{C}$, is obtained as :

$$\mathbf{R} = \mathbf{R}_\mathcal{M}^\top(\mathbf{R}^\mathcal{M}\mathbf{R}_x^\mathcal{M})\mathbf{R}_\mathcal{M}$$

- The **translation** $\mathbf{t}$ is then obtained by back-substituting $\mathbf{R}$ into the linear system $\mathbf{n}^\top(\mathbf{R}\mathbf{X} + \mathbf{t}) = 0$ which can be solved by SVD decomposition. The solver might have several solutions, it is choosing the geometrically valid solution

In case of a **multi-view camera system**, we solve the absolute pose $(\mathbf{R}, \mathbf{t})$ for the reference camera, then the **relative pose** $(\mathbf{R}_i, \mathbf{t}_i)$ of the other cameras can be solved using the step in points 1-5 by applying $(\mathbf{R}, \mathbf{t})$ to the 3D data

Pose Refinement

We formulate a least-squares refinement for the **multi-view** case based on the initial equations.

$$\begin{aligned} &\forall j = 1, \dots, N_{\mathcal{C}_0} : \\ &\quad \mathbf{n}_j^{\mathcal{C}_0\top}\mathbf{R}\mathbf{V}_j = 0 \\ &\quad \mathbf{n}_j^{\mathcal{C}_0\top}(\mathbf{R}\mathbf{X}_j + \mathbf{t}) = 0 \\ &\forall i = 1, \dots, M-1; \forall j = 1, \dots, N_{\mathcal{C}_i} : \\ &\quad \mathbf{n}_j^{\mathcal{C}_i\top}\mathbf{R}_i\mathbf{R}\mathbf{V}_j = 0 \\ &\quad \mathbf{n}_j^{\mathcal{C}_i\top}(\mathbf{R}_i(\mathbf{R}\mathbf{X}_j + \mathbf{t}) + \mathbf{t}_i) = 0 \end{aligned}$$

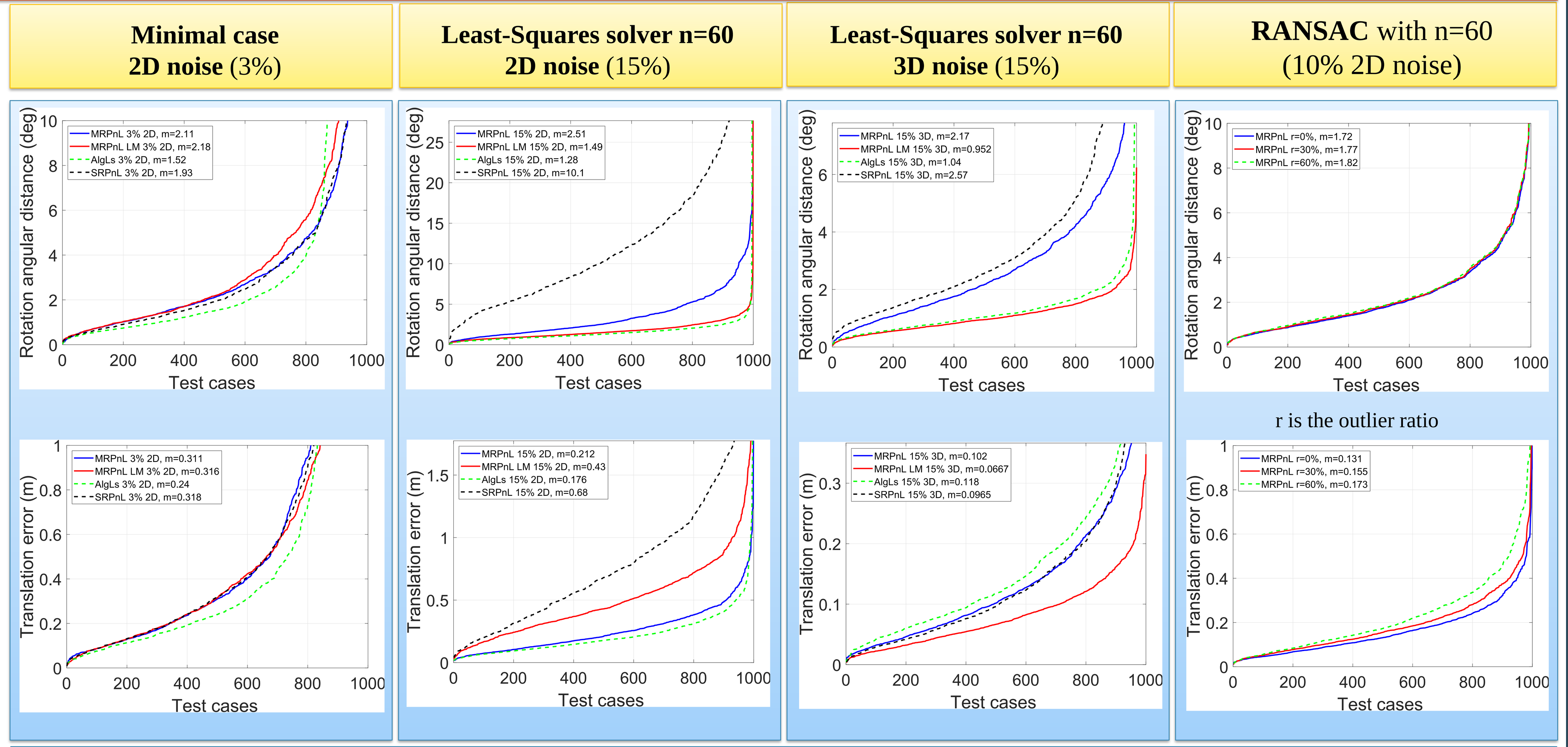
Solved via standard algorithms like Levenberg-Marquardt with the initialization obtained from the direct solver.

We used the automatic generator of Kukelova et al. [2] for a fair comparison in Matlab with competing methods, we also used Kneip's generator [3] to produce a solver in C++

Evaluation on Synthetic Data

- ❖ Benchmark dataset of 1000 2D-3D synthetic images
- ❖ **3D scene**: 3 arbitrary planes with 20 lines on each
- ❖ **2D side**: we generated images of the scene by projecting the lines on perspective cameras with 2378x1580 resolution and real parameters
- ❖ **Noise**: Corrupting one endpoint of the line (similarly in 2D and 3D), essentially adding a random number to each coordinate of the point
  - ❖ 3%, 10% and 15% in 2D and 3D

Synthetic Data Plots



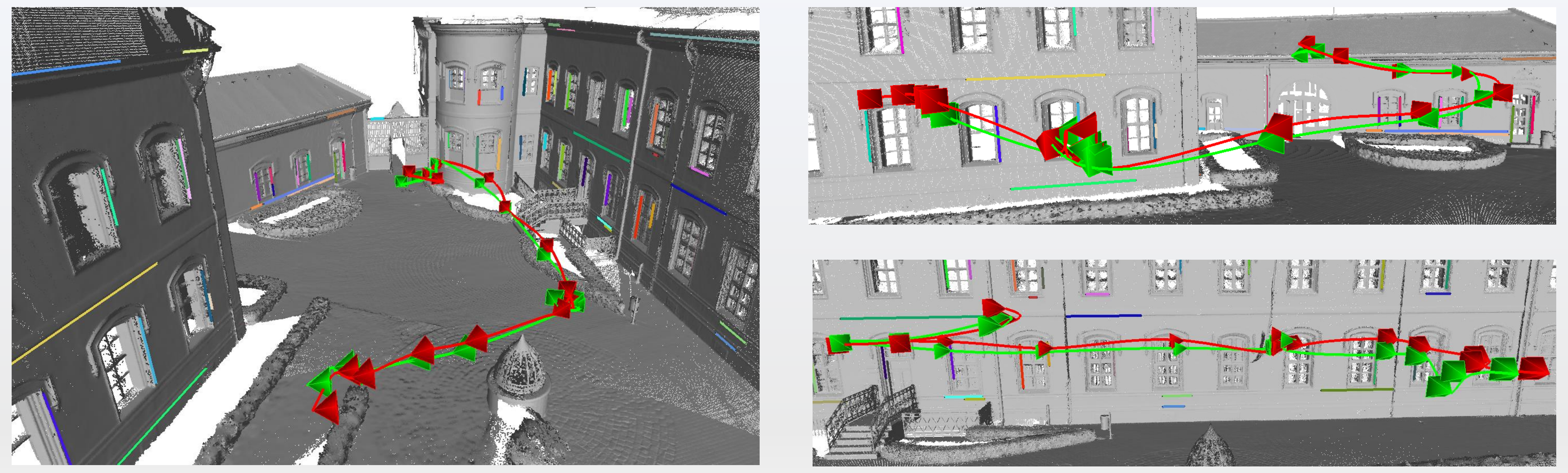
	MRPnL	MRPnL LM	ALgLS	SRPnL
Minimal case	0.0022	0.0083	0.0536	0.0036
60 lines	0.0026	0.009	0.0597	0.0030

Runtime in seconds, median over 1000 test cases with 2D noise (ALgLS requires at least 4 lines)

RANSAC: while a higher than 50% outlier ratio can be filtered out robustly, it drastically increases the execution time:

- with 30% outlier ratio is 0.16 (s)
- with 60% outlier ratio reaches 1.94 (s)

Real Data



MRPnL-LM **trajectory estimation results** on 16 frames of a longer **drone sequence.** Ground truth camera poses and the trajectory are shown in green, the estimated ones in red, while the used 3D lines (81 in total) are also visible.

References

[1] G. H. Lee. A minimal solution for non-perspective pose estimation from line correspondences. **ECCV-2016**. Springer
[2] Z. Kukelova et al. Automatic generator of minimal problem solvers. **ECCV-2008**. Springer.
[3] L. Kneip. Polyjam, 2015 [online]. url:https://github.com/laurentkneip/polyjam.

Paper

Acknowledgements

This work was partially supported by the NKFI-6 fund through project K120366; "Integrated program for training new generation of scientists in the fields of computer science", EFOP-3.6.3-VEKOP-16-2017-0002; the Research & Development Operational Programme for the project "Modernization and Improvement of Technical Infrastructure for Research and Development of J. Selye University in the Fields of Nanotechnology and Intelligent Space", ITMS 26210120042, co-funded by the European Regional Development Fund; Research & Innovation Operational Programme for the Project: "Support of research and development activities of J. Selye University in the field of Digital Slovakia and creative industry", ITMS code: NFP313010T504, co-funded by the European Regional Development Fund.

Research Group