



UNIVERSITY OF ZAGREB

Faculty of Electrical
Engineering and
Computing

3D Computer Vision

Robust motion estimation

Ivan Marković

University of Zagreb Faculty of Electrical Engineering and Computing
Department of Control and Computer Engineering
Laboratory for Autonomous Systems and Mobile Robotics (lamor.fer.hr)

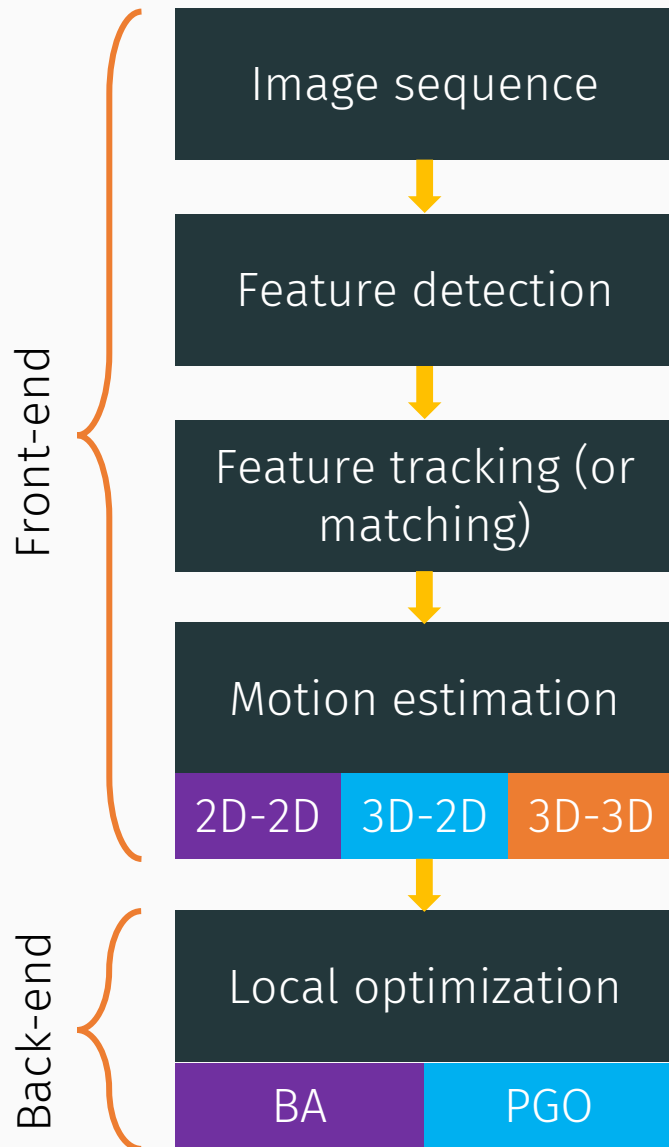
Outline

- Introduction
- RANSAC in a nutshell
- RANSAC and Visual odometry
- Reducing the minimal number of correspondences
 - Planar motion (2-points RANSAC)
 - Planar circular motion (1-point RANSAC)
- Odometry evaluation

Outline

- Introduction
- RANSAC in a nutshell
- RANSAC and Visual odometry
- Reducing the minimal number of correspondences
 - Planar motion (2-points RANSAC)
 - Planar circular motion (1-point RANSAC)
- Odometry evaluation

Odometry with real-world data



Real world is riddled with uncertainty, noise, incorrect feature matches, occlusions etc. This introduces a fair number of outliers in the data and can significantly affect the visual odometry accuracy.

The **motion estimation** block usually involves a robust model estimation procedure such as **RANSAC** that is used to determine the set of inliers based on which the final motion parameters are computed.

Outline

- Introduction
- RANSAC in a nutshell
- RANSAC and Visual odometry
- Reducing the minimal number of correspondences
 - Planar motion (2-points RANSAC)
 - Planar circular motion (1-point RANSAC)
- Odometry evaluation

Getting robust with RANSAC

To have robust visual odometry, we need to be able to deal with **outliers**.

Most commonly, **RANSAC** (introduced in the 3D reconstruction part of the course) is used for this purpose.

RANSAC is the standard method for model fitting in the presence of outliers (even **more than 90%!**).

It is **non-deterministic**: you get a different result every time you run it; however, it is not sensitive to the initial condition, and does not get stuck in local maxima like nonlinear optimization approaches.

It is very diverse and can be applied to all sorts of problems.

Getting robust with RANSAC

A straightforward approach to RANSAC would try out all the possible combinations of points, e.g., in line fitting this would entail all possible pairs of points.

If we had 1000 points, we would need to try out $N(N-1)/2$ combinations, i.e., close to 500.000, which can be computationally infeasible.

However, RANSAC¹ approaches this in a probabilistic way – if we have a rough estimate of the percentage of inliers in our dataset, we can only check a subset of all combinations, i.e., perform a certain number of iterations, and claim results with a certain probability of success.

Specifically, it was shown that the number of iterations amounts to

$$k = \frac{\log(1 - p)}{\log(1 - w^s)}$$

The diagram shows the formula $k = \frac{\log(1 - p)}{\log(1 - w^s)}$ with several annotations: a blue arrow points from the text 'Ratio of inliers' to the variable w in the denominator; a purple arrow points from the text 'Probability of success' to the variable p in the numerator; and an orange arrow points from the text 'Number of points for the model' to the variable s in the denominator.

¹M. A. Fischler, R. C. Bolles (1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography."

RANSAC model fitting

General pseudocode of RANSAC applied to general model fitting would be:

1. Repeat
2. **Randomly** select a sample of m points
3. Fit the **model** using the selected m points
4. Compute the **distances** d of all the other points from this model
5. Construct the **inlier set** (points with d smaller than threshold)
6. Store these inliers
7. Until maximum number of iterations k

The set with the maximum number of inliers is chosen as the solution to the problem. For the line fitting example, with $w=0.5$ and $p=99\%$ we would need only $k=16$ iterations (vs. exhaustive 500.000). Also, note that the total number of points does not influence k .

Outline

- Introduction
- RANSAC in a nutshell
- RANSAC and Visual odometry
- Reducing the minimal number of correspondences
 - Planar motion (2-points RANSAC)
 - Planar circular motion (1-point RANSAC)
- Odometry evaluation

Applying RANSAC to VO

For the case of applying RANSAC to visual odometry, the questions are:

1. What is our model in visual odometry?
2. What is the minimum number of required points to estimate the model?
3. How to compute the distances from the model, i.e., how to construct a metric that measures how well a point fits the model?

The beauty of RANSAC is that it can be applied to a diverse set of problems, thus we can use it within the 3D-3D, 3D-2D and 2D-2D motion estimation approaches.

The three VO groups and RANSAC

For the **3D-3D motion estimation** we would have:

1. The motion rotation matrix and the translation vector
2. 3 non-collinear points
3. Distance to 3D correspondences

For the **3D-2D motion estimation**:

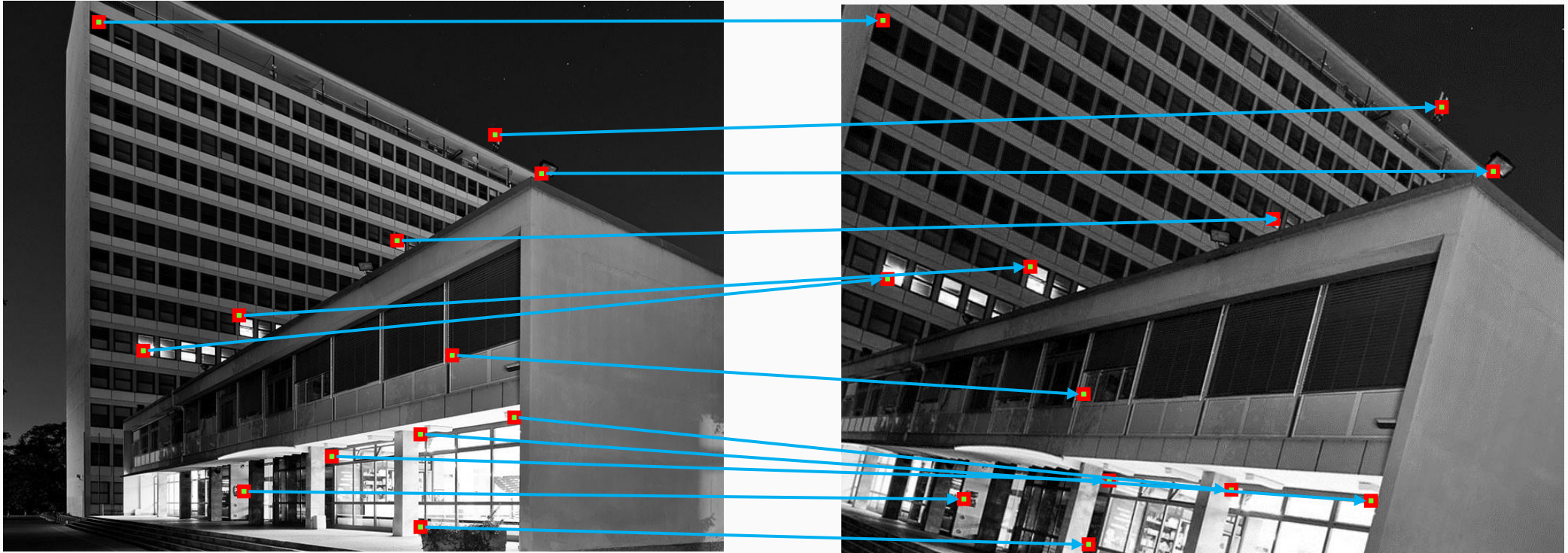
1. The motion rotation matrix and the translation vector
2. 3 non-collinear points (+1 for disambiguation)
3. Reprojection error

For the **2D-2D motion estimation** we would have:

1. The essential matrix
2. 5 points for Nister and 8 points for the Longuet-Higgins algorithm
3. Reprojection error, distance to epipolar lines ...

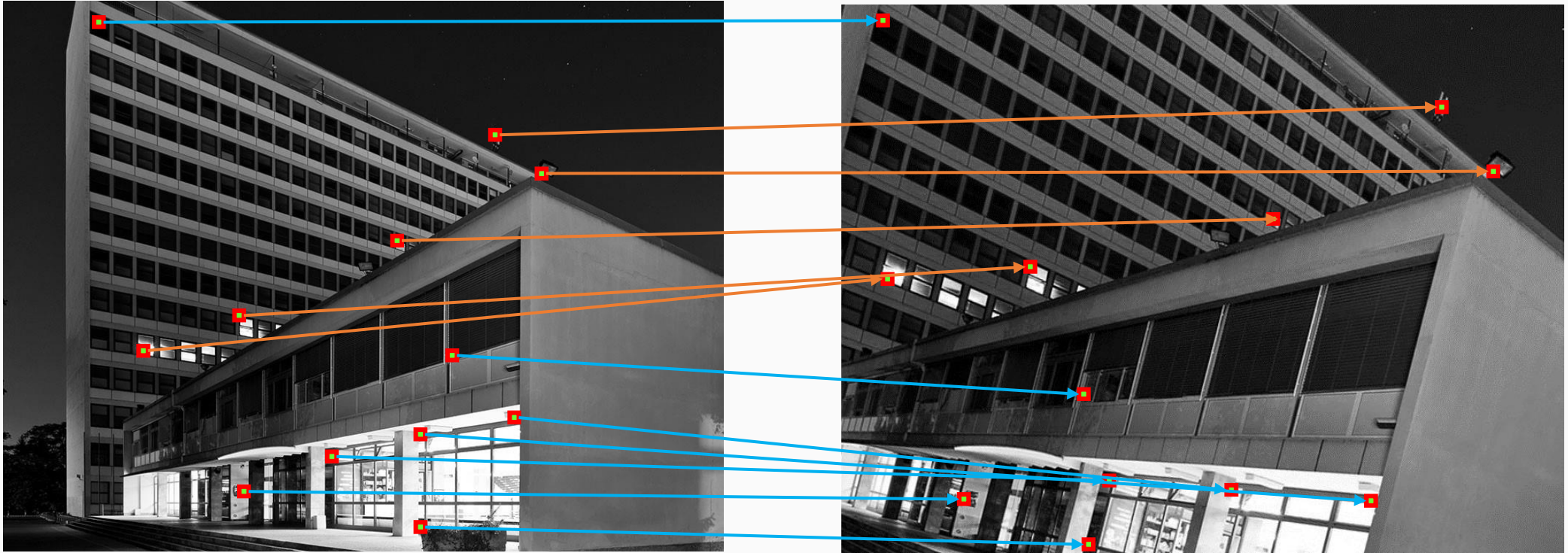
5-point RANSAC

Illustration of running RANSAC with the 5-pt algorithm.



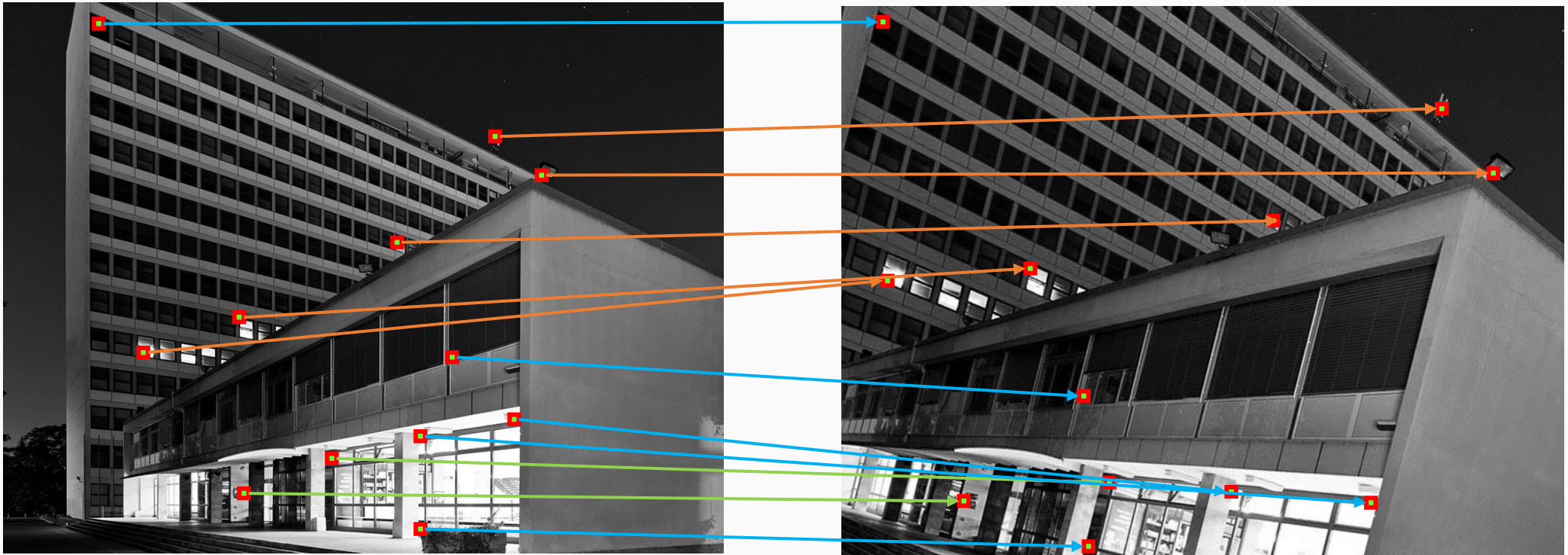
5-point RANSAC

Randomly select 5 points (orange).



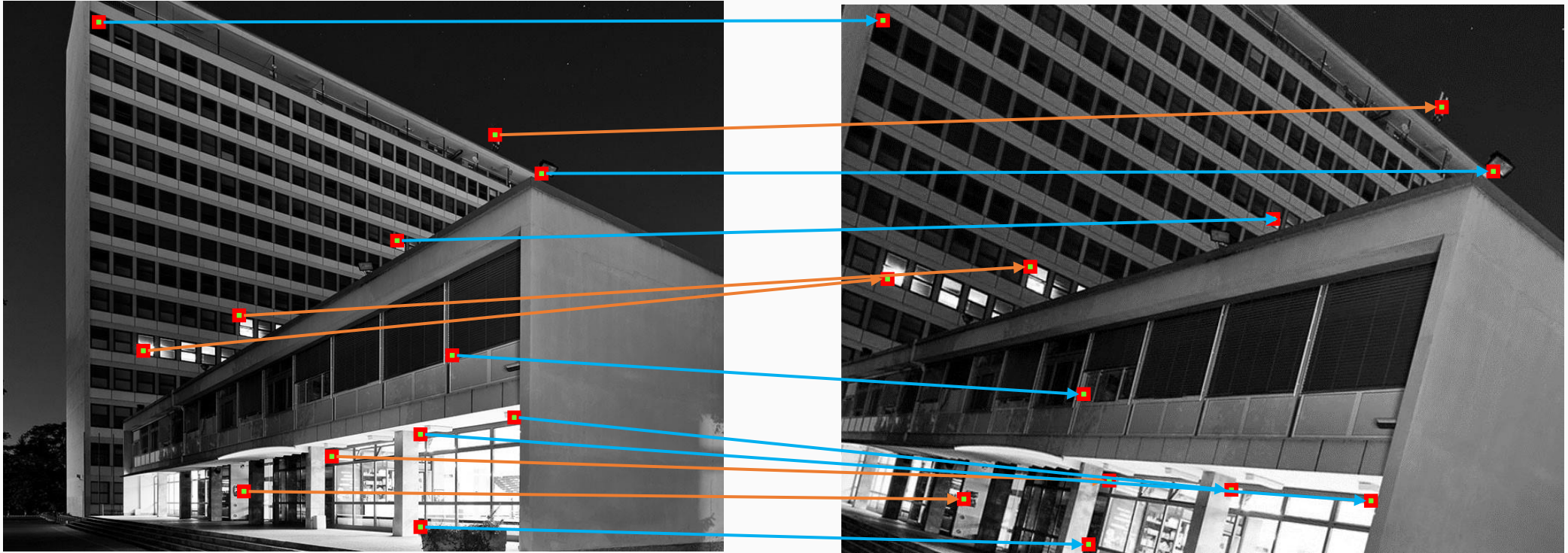
5-point RANSAC

Compute the model and find inliers (green) by thresholding $x^T E x' < \tau$.



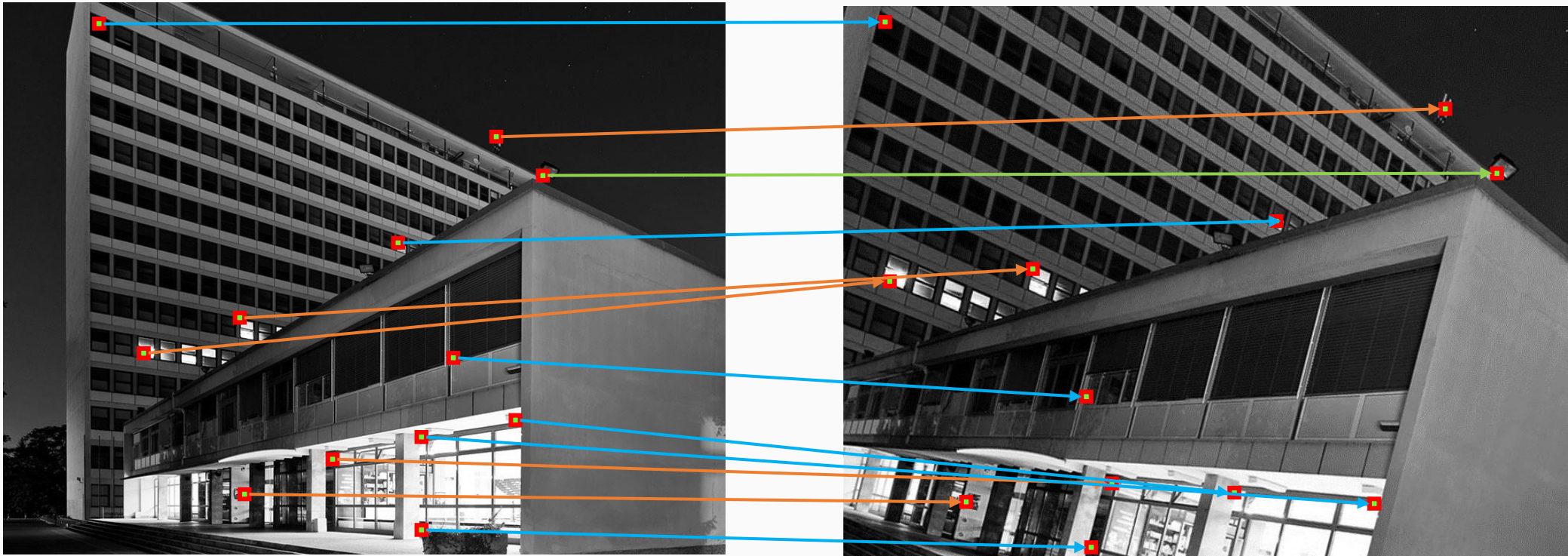
5-point RANSAC

Randomly select 5 points (orange).



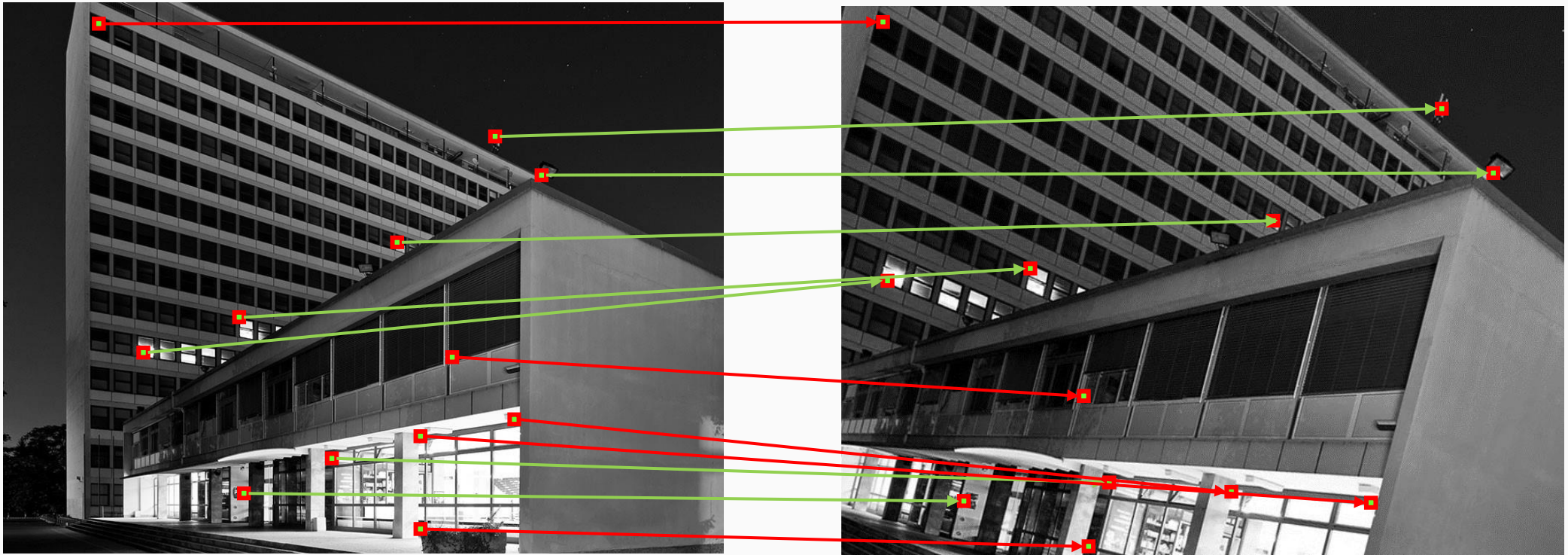
5-point RANSAC

Compute the model and find inliers (green) by thresholding $x^T E x' < \tau$.



5-point RANSAC

The model with the highest number of inliers wins.



Note that the Níster's algorithm, unlike the 8-point algorithm, can only use 5 points to compute the essential matrix; no more, no less.

RANSAC #iterations

The number of required iterations **increases exponentially** with the number of points needed to estimate the model.

If we assume $p=99\%$ and $w=50\%$ then:

- 8-point RANSAC

$$k = \frac{\log(1 - 0.99)}{\log(1 - 0.5^8)} = 1177 \text{ iterations.}$$

Requires more iterations, can use more than 8 points to compute the essential matrix, not directly in space of essential matrices = finds the „closest“ essential matrix, but returns unique solution.

- 5-point RANSAC

$$k = \frac{\log(1 - 0.99)}{\log(1 - 0.5^5)} = 145 \text{ iterations.}$$

Requires less iterations, can only use 5 points to compute the essential matrix, returns up to 10 solutions (worst case) in the essential matrix space.

- 2-point RANSAC (for line fitting, not essential matrix 😊)

$$k = \frac{\log(1 - 0.99)}{\log(1 - 0.5^2)} = 16 \text{ iterations.}$$

Outline

- Introduction
- RANSAC in a nutshell
- RANSAC and Visual odometry
- Reducing the minimal number of correspondences
 - Planar motion (2-points RANSAC)
 - Planar circular motion (1-point RANSAC)
- Odometry evaluation

Introducing motion constraints

The **lower the number of points** necessary to estimate the model the better, since RANSAC will require **fewer iterations** to find the inlier set.

We know that a minimum of 5 points is needed to estimate the essential matrix, but is it possible to use less than 5 points in certain instances?

The 5-point algorithm estimates 6DoF camera motion; however, if the camera cannot exhibit motion in all the DoF, e.g., if it is mounted on a platform that moves only in 2D, then the motion constraints can be taken into account to reduce the minimum number of required points.

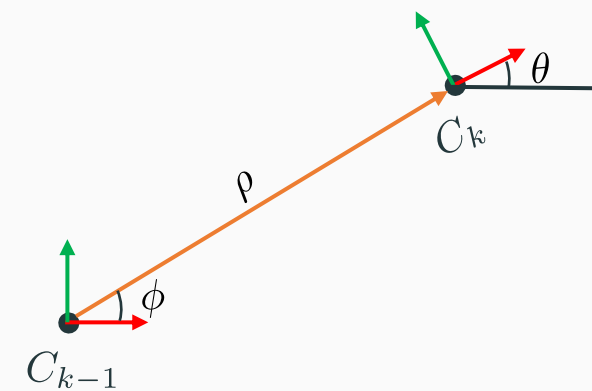
Also, if sensor fusion is performed and another sensor is used to solve for some DoF, e.g., an accelerometer that can measure the gravity direction, the number of required points can also be reduced (e.g, to three correspondences)¹.

¹ O. Naroditsky, X. S. Zhou, S. I. Roumeliotis, K. Daniilidis (2012). „Two Efficient Solutions for Visual Odometry Using Directional Correspondence.”

Planar motion

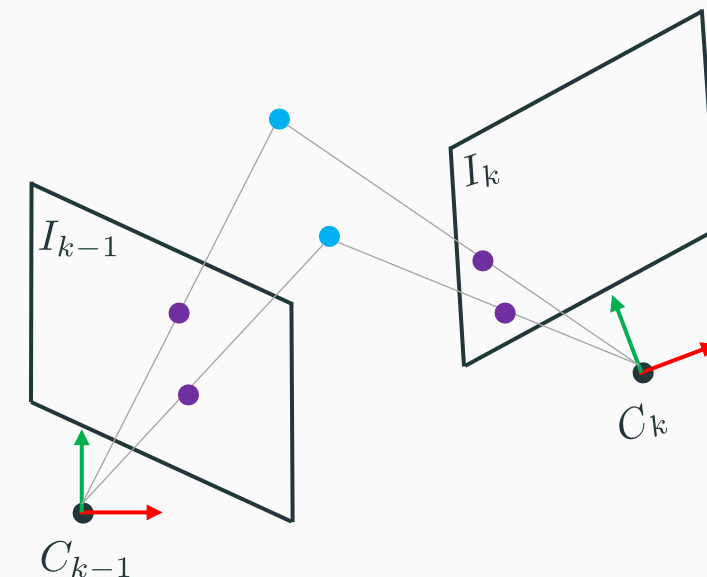
Planar motion can be described with 3 parameters (rotation around the z axis and translation with zero z component)

$$R = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad t = \begin{bmatrix} \rho \cos \phi \\ \rho \sin \phi \\ 0 \end{bmatrix}$$



In this case the essential matrix has **only 2DoF** and only 2 correspondences are needed (see the **Board**)²

$$E = [t]_{\times} R = \begin{bmatrix} 0 & 0 & \rho \sin \phi \\ 0 & 0 & -\rho \cos \phi \\ -\sin(\phi - \theta) & \rho \cos(\phi - \theta) & 0 \end{bmatrix}.$$



² D. Ortín and J. M. M. Montiel (2001). „Indoor robot motion based on monocular images.”

Planar motion epipolar constraint

In this case for a single correspondence the epipolar constraint amounts to

$$E = p_2^T E p_1 = -x_1 \sin \theta_1 + x_2 \sin \theta_2 + y_1 \cos \theta_1 - y_2 \cos \theta_2 = 0,$$

where $\theta_1 = \phi - \theta$, $\theta_2 = \phi$.

A system of equations can be written in the form of $Ax = 0$ with unknowns being $x = [\cos \theta_1 \quad \sin \theta_1 \quad \cos \theta_2 \quad \sin \theta_2]$; however, the elements of x are not independent, i.e., $\sin^2 \theta_1 + \cos^2 \theta_1 = 1$, thus only **2 points correspondences** (2 quadratic equations) are needed.

Given that, algebraically there are 4 possible solutions, out of which in general, two will allow for negative depths and can be discarded.

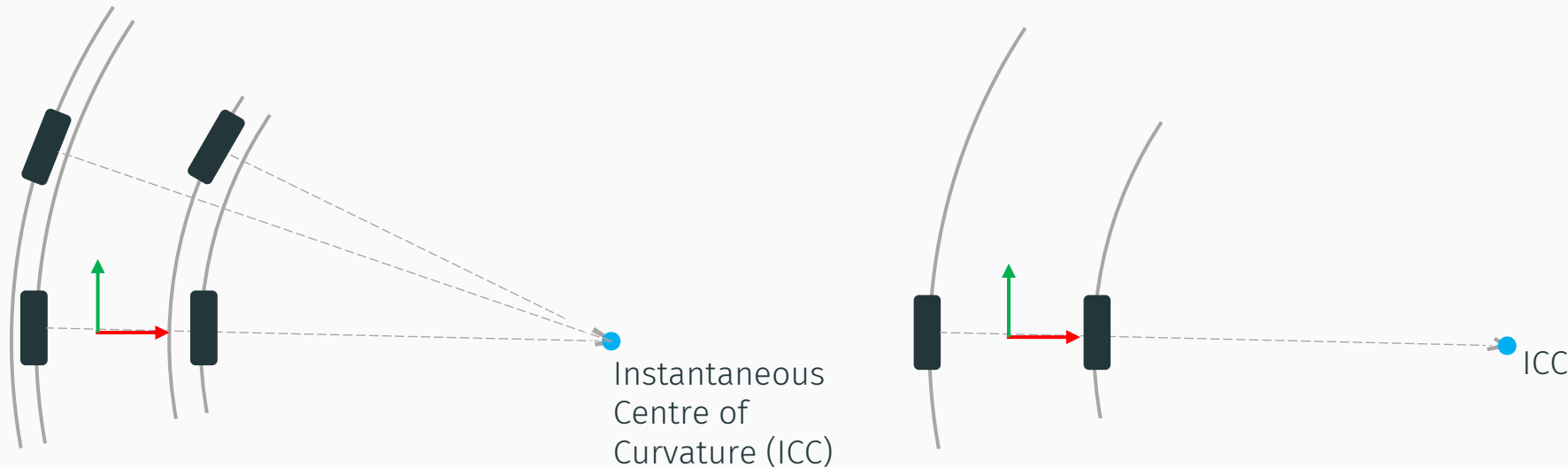
Alternatively, a nonlinear optimization with an initial guess can be used to reach the final solution, e.g., Gauss-Newton.

Planar circular motion

Can we go lower than 2 points? Yes, in case of **planar circular motion**.

This is not uncommon when the camera is placed on specific mobile robot configurations or vehicles, since their motion is locally circular.

For examples, vehicles with the Ackermann drive (only if the camera is placed above the rear axle; otherwise, use an approximation³) or differential drive mobile robots exhibit circular motion.



³ D. Scaramuzza (2011). „1 Point RANSAC Structure from Motion for Vehicle Mounted Cameras by Exploiting Non-Holonomic Constraints”

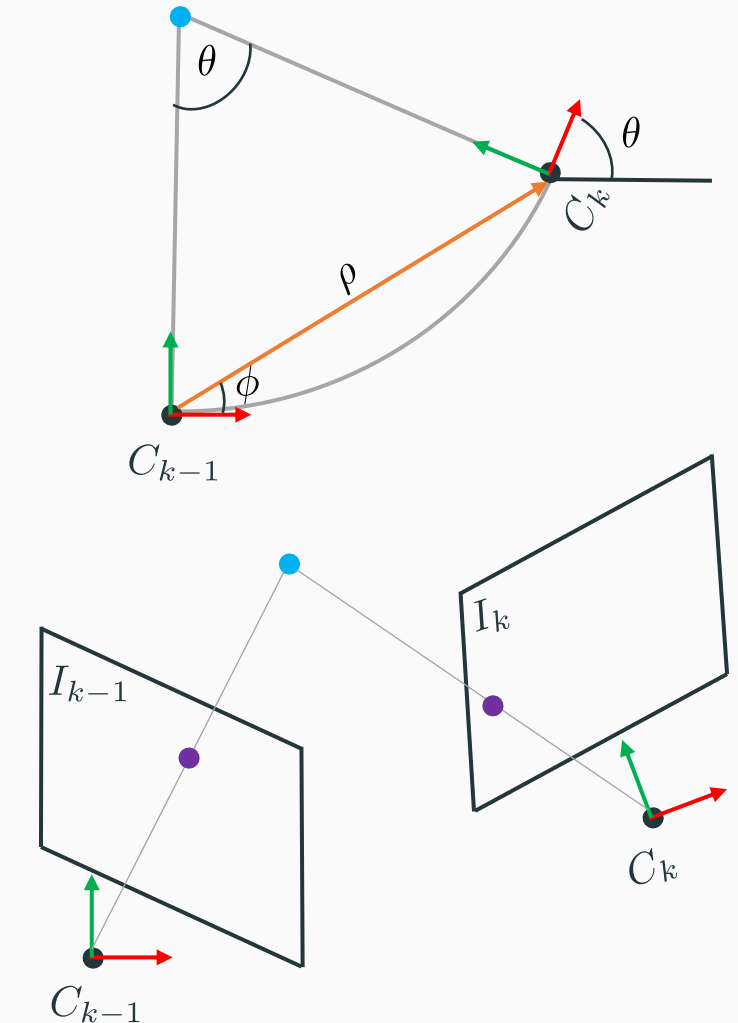
Planar circular motion

In the case of planar circular motion the camera undergoes motion along the circle and the local x axis is tangent to the circle, while the y axis points in the ICC direction, i.e., the circle's origin. In this case it can be shown (see the **Board**) that

$$\phi = \frac{\theta}{2},$$

Which means that we have **only 1 DoF** (ρ is still the scale factor) and only a **single feature correspondence** is necessary.

This is the **smallest parameterization possible** and results in the most efficient algorithm for removing outliers³.



³ D. Scaramuzza (2011). „1 Point RANSAC Structure from Motion for Vehicle Mounted Cameras by Exploiting Non-Holonomic Constraints”

Planar circular motion epipolar constraint

In this case the rotation and translation evaluate to

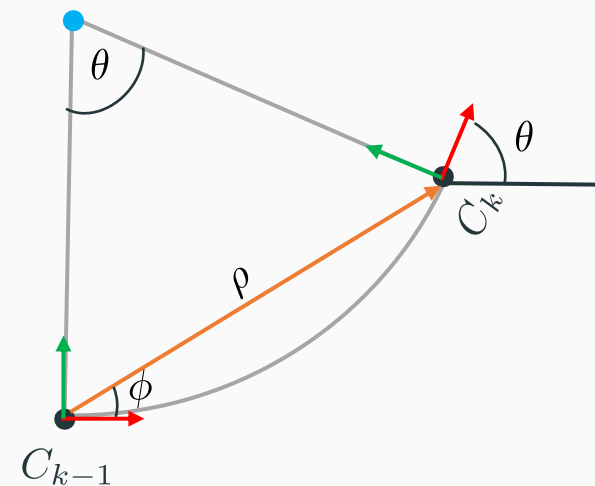
$$R = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad t = \begin{bmatrix} \rho \cos \frac{\theta}{2} \\ \rho \sin \frac{\theta}{2} \\ 0 \end{bmatrix}$$

The essential matrix then evaluates to (see the **Board**)

$$E = [t]_{\times} R = \begin{bmatrix} 0 & 0 & \rho \sin \frac{\theta}{2} \\ 0 & 0 & -\rho \cos \frac{\theta}{2} \\ \rho \sin \frac{\theta}{2} & \rho \cos \frac{\theta}{2} & 0 \end{bmatrix},$$

and the epipolar constraint is

$$E = p_2^T E p_1 = (x_1 + x_2) \sin \frac{\theta}{2} - (y_2 - y_1) \cos \frac{\theta}{2} = 0, \Rightarrow \theta = 2 \tan^{-1} \left(\frac{y_2 - y_1}{x_1 + x_2} \right).$$



1-point RANSAC

The **1-point RANSAC** randomly selects a single feature correspondence, computes θ , and then the corresponding rotation and translation (ρ can be arbitrarily set to 1).

The motion hypothesis is carried out and the inlier set is found by computing some distance metric, e.g., the reprojection error.

If we assume $p=99\%$ and $w=50\%$ then

$$k = \frac{\log(1 - 0.99)}{\log(1 - 0.5^1)} = 7 \text{ iterations.}$$

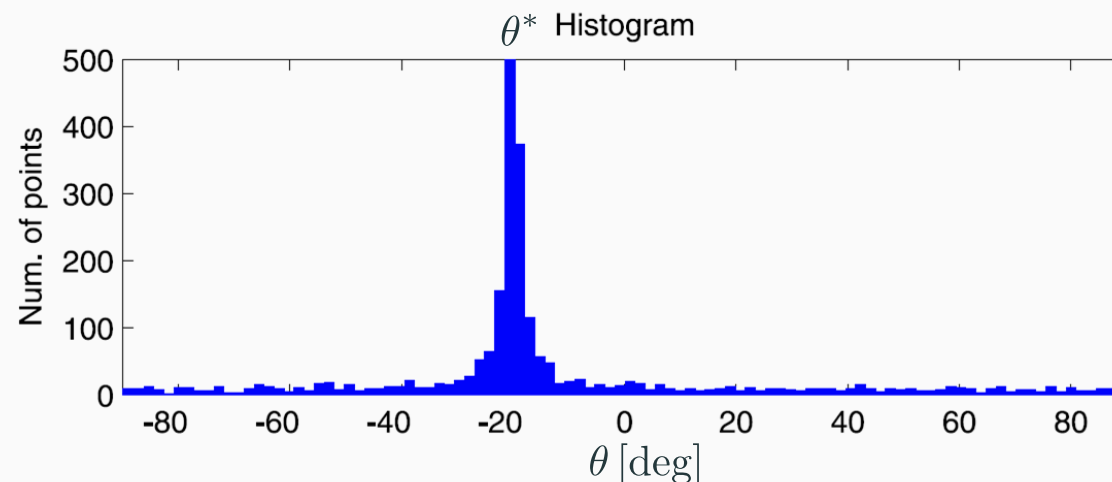
The possibility of estimating the motion using only one feature correspondence allows us to implement another algorithm for outlier removal which is much more efficient than the 1-point RANSAC as it requires no iterations.

Histogram voting

The algorithm is based on **histogram voting**: first, θ is computed from each feature correspondence using (10); then, a histogram H is built where each bin contains the number of features which count for the same θ .

When the circular motion model is well satisfied, the histogram has a very narrow peak centered on the best motion estimate θ^* .

We generate the motion hypothesis by substituting θ^* and compute the rotation and translation, which are then used to compute the reprojection error and identify all the inliers.



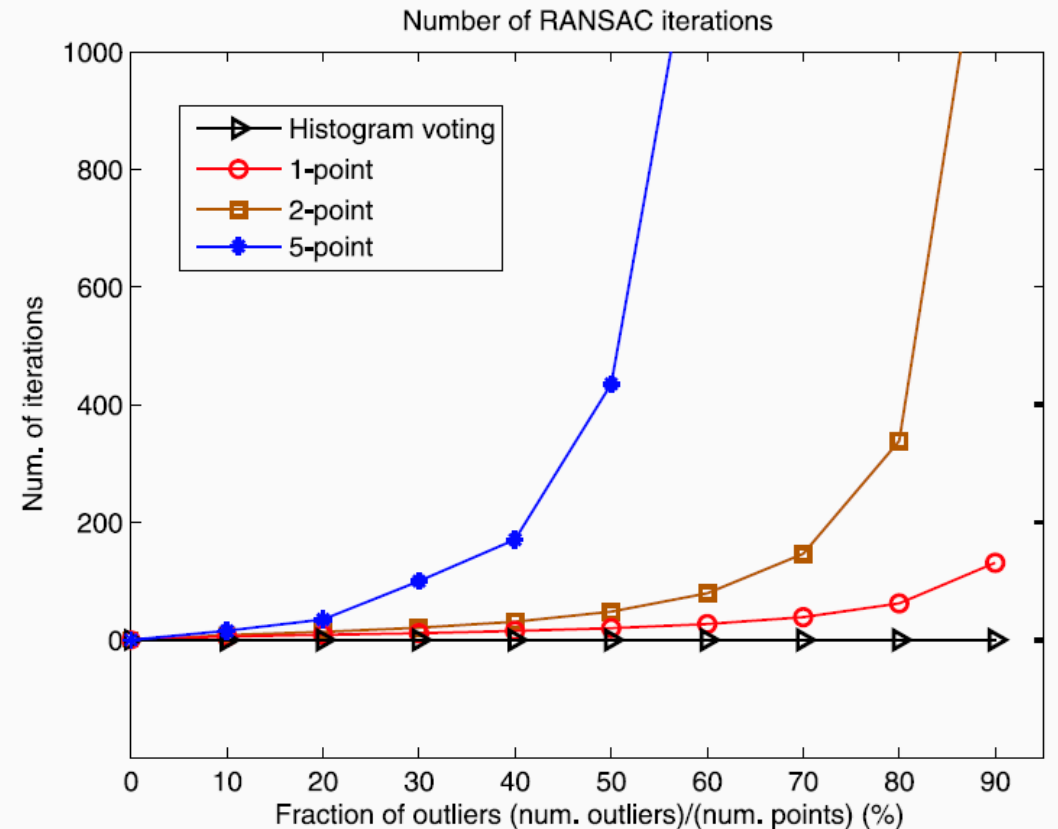
*D. Scaramuzza (2011). „1 Point RANSAC Structure from Motion for Vehicle Mounted Cameras by Exploiting Non-Holonomic Constraints”

RANSAC comparison

Note that the 1-point algorithms (either RASNAC or histogram voting) are **only used to find the set of inliers**. The 6DoF motion is then computed using standard methods (5- or 8-point algorithm).

Authors report for the histogram voting an average of 0.2 ms with a dataset of about 1600 points. The 1-point RANSAC finds a successful solution in less than 7 iterations, requiring at most 1 ms.

Note that complexity increases exponentially with the the fraction of outliers.



*D. Scaramuzza (2011). „1 Point RANSAC Structure from Motion for Vehicle Mounted Cameras by Exploiting Non-Holonomic Constraints”

Conclusion

When dealing with real-world data it is necessary to use **robust estimation methods** as sensor noise and environmental effects will cause quite a bit of incorrect correspondences (outliers).

To identify the set of inliers, commonly **RANSAC** is used.

For unconstrained motion RANSAC is coupled with standard 5- or 8-points algorithms; however, when the camera exhibits only planar motion or even circular planar motion, the necessary number of feature correspondences can be further reduced.

In the case of **planar motion**, only **2 points** correspondences are sufficient, while for **planar circular motion** only **1 point** correspondence is sufficient (enabling highly efficient histogram voting for inlier detection).

Finally, with the identified inlier set, standard motion estimation methods can then be applied.

Outline

- Introduction
- RANSAC in a nutshell
- RANSAC and Visual odometry
- Reducing the minimal number of correspondences
 - Planar motion (2-points RANSAC)
 - Planar circular motion (1-point RANSAC)
- Odometry evaluation

Trajectory evaluation dilemma

How can we evaluate and compare visual odometry algorithms? Two subquestion arise:

1. what should the solution be compared to?
2. on which datasets should we compare the algorithms?
3. which metric should we use?

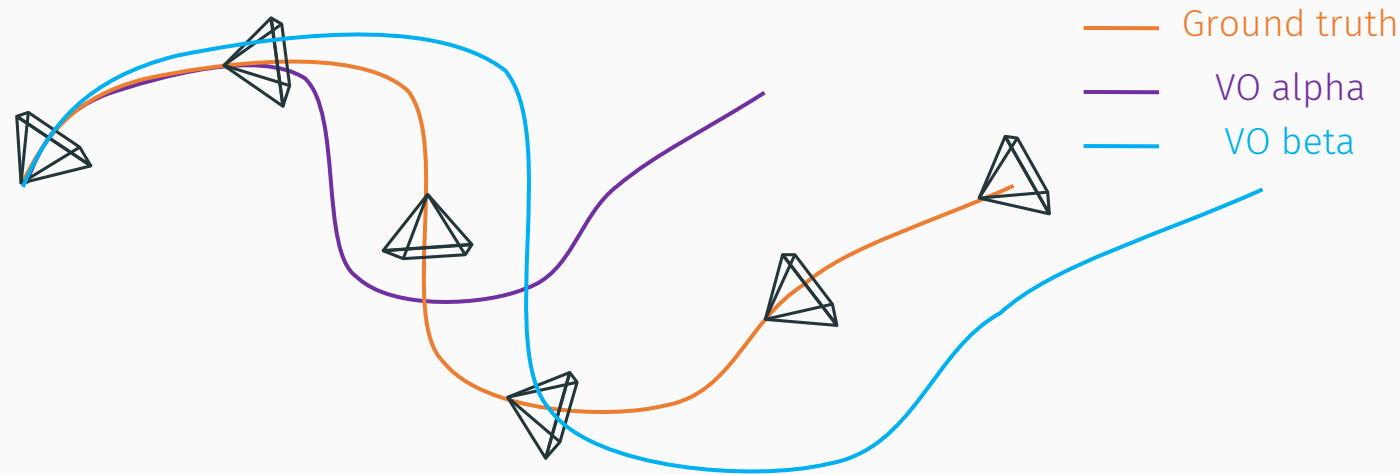
Ad 1. We need **ground truth trajectory** obtained using a highly accurate sensor or fusion thereof. For example, RTK GPS fused with highly accurate IMU, or a 3D laser range sensor SLAM trajectory, or fusion of both.

Ad 2. Choose datasets commonly used in state of the art, possibly with online benchmarking.

Ad 3. Lets look at few examples first.

Trajectory evaluation dilemma

Example below shows the ground truth trajectory and two different odometry solutions: one is underscaled and the other is overscaled. The first frame poses have been aligned.



Which one is better?

Trajectory evaluation dilemma

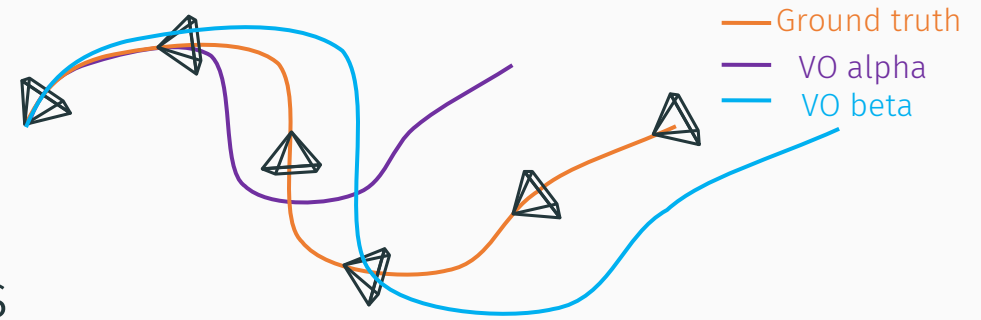
Actually, both are equally good/bad. One is 25% underscaled, while the other is 25% overscaled.

We might focus on the end-point error, but this does not take into account the whole trajectory shape.

Should we align the first, middle, or the final camera pose? Should we compare frame-to-frame camera pose errors with exact same timestamps?

Additionally, should we approach differently evaluation of odometry that has metric scale (e.g., stereo visual odometry or 3D laser odometry) compared to an up-to-scale odometry (e.g., monocular visual odometry)?

In literature two main approaches are used: absolute trajectory error (ATE) and relative pose error (RPE).



Absolute trajectory error

Absolute trajectory error (ATE) consists of two steps:

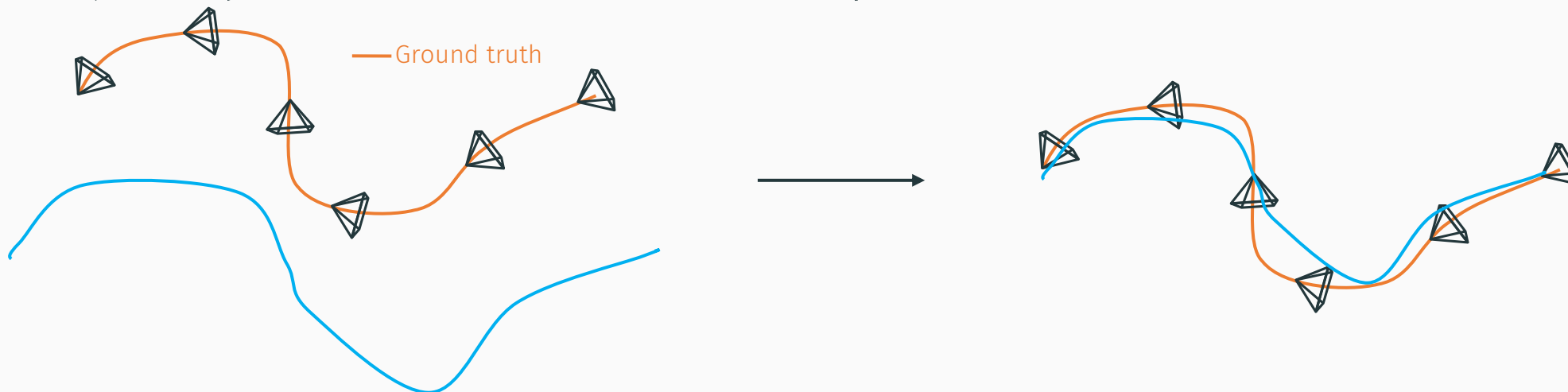
1. Align the whole estimated trajectory with the ground truth by minimizing the sum of squared pose errors.

$$(R^*, t^*, \lambda^*) = \arg \min_{R, t, \lambda} \sum_{k=1}^m \|\hat{C}_k - (\lambda RC_k + t)\|^2.$$

Ground truth
camera poses

Scale factor

Note that we align the orientation and translation as well as the trajectory scale over all the camera poses.



2. After alignment the root mean squared error is computed

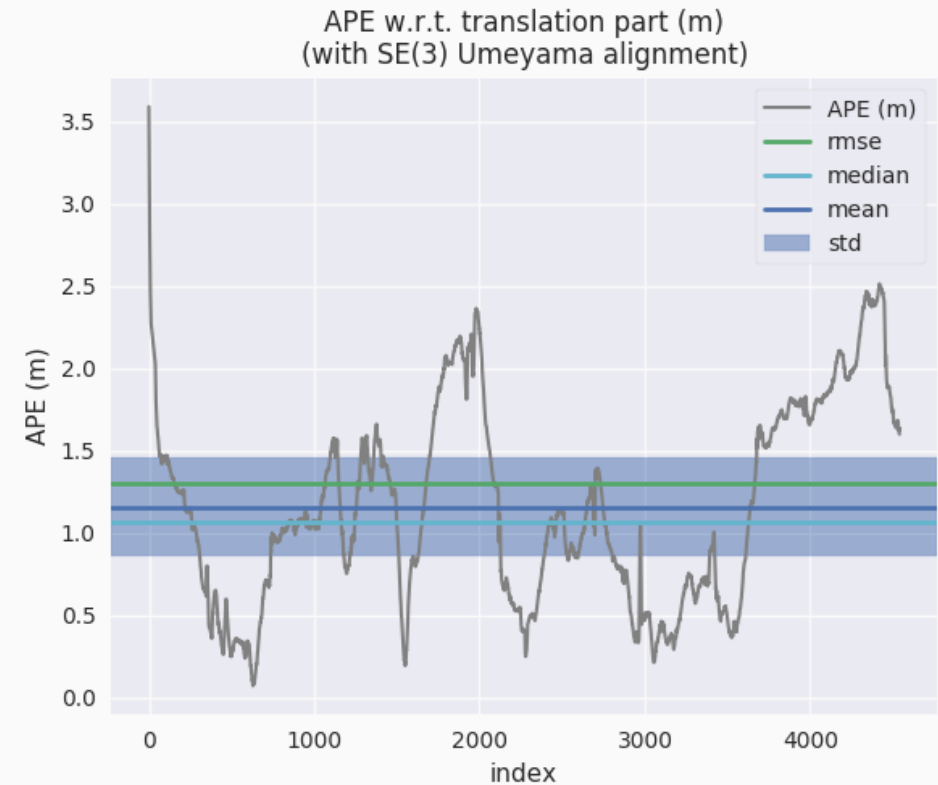
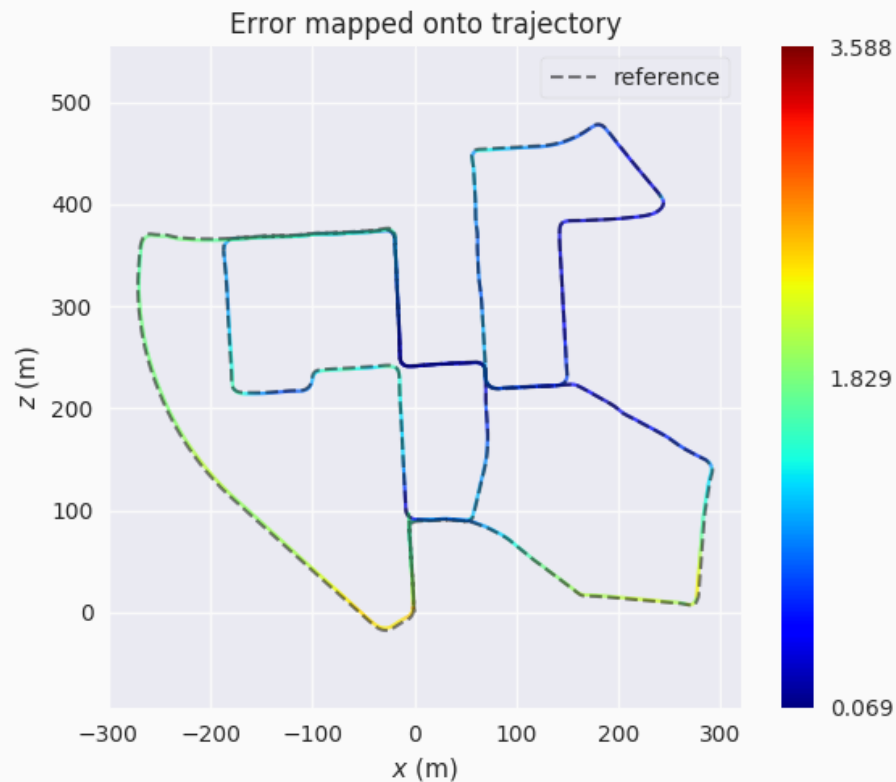
$$\text{ATE} = \sqrt{\sum_{k=1}^m \|\hat{C}_k - (\lambda^* R^* C_k + t^*)\|^2}.$$

The advantage of ATE is that it produces a single number based on which odometries can be compared and it captures the global trajectory error (making it a suitable metric for SfM and SLAM approaches).

However, it does not capture the relative error, i.e., the local trajectory estimates. Should the odometry be strictly penalized if after a couple of turns it had a few degree error but after that performed flawlessly?

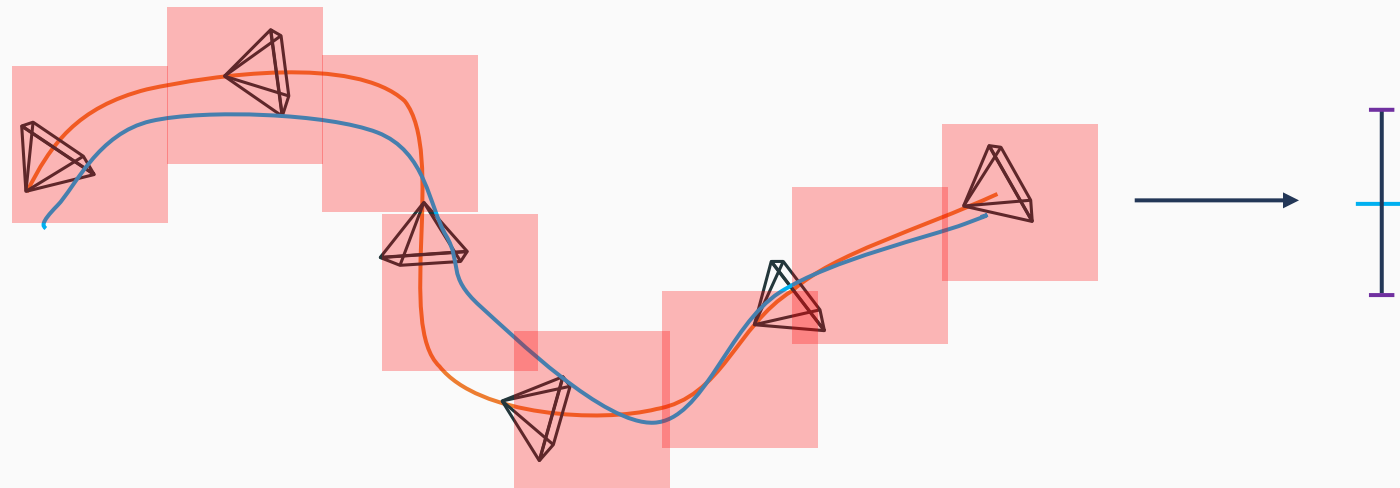
Absolute trajectory error

An example of how an open-source package [evo](#) computes and visualizes the odometry performance is depicted below.



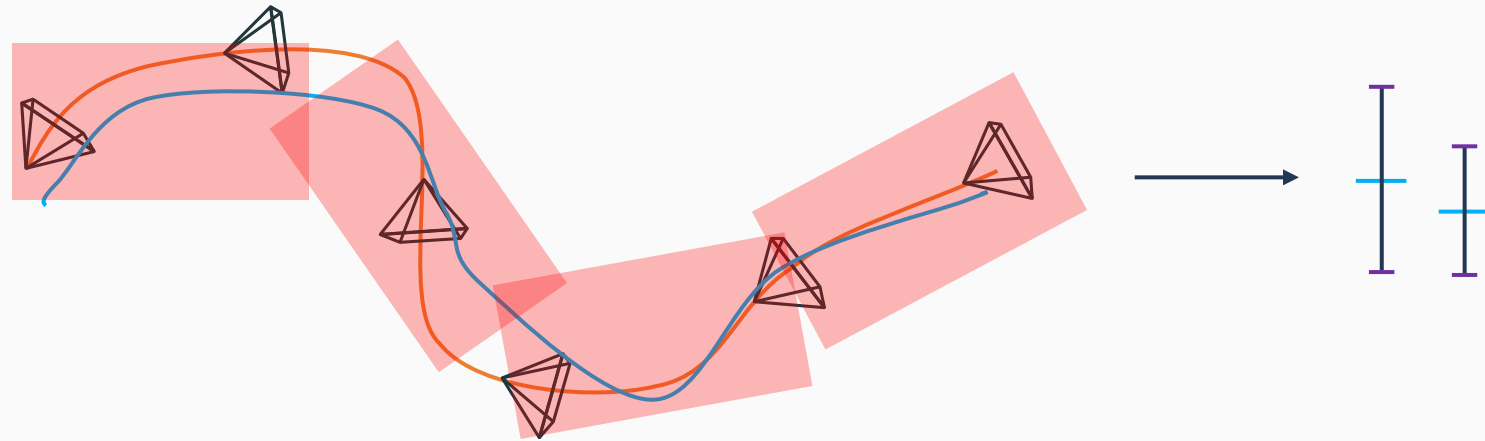
Relative pose error

This is where **relative pose error** (RPE) steps in. The idea of RPE is to evaluate the odometry based on computing error statistics of sub-trajectories of specific lengths.



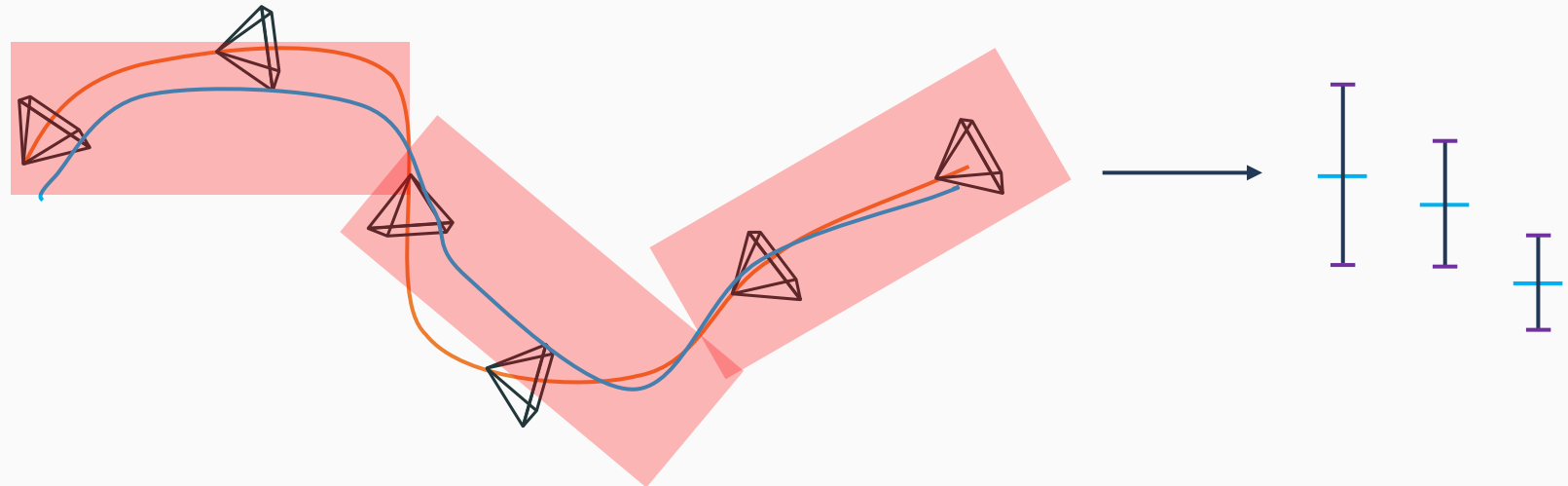
Relative pose error

This is where **relative pose error** (RPE) steps in. The idea of RPE is to evaluate the odometry based on computing error statistics of sub-trajectories of specific lengths.



Relative pose error

This is where **relative pose error** (RPE) steps in. The idea of RPE is to evaluate the odometry based on computing error statistics of sub-trajectories of specific lengths.



Relative pose error

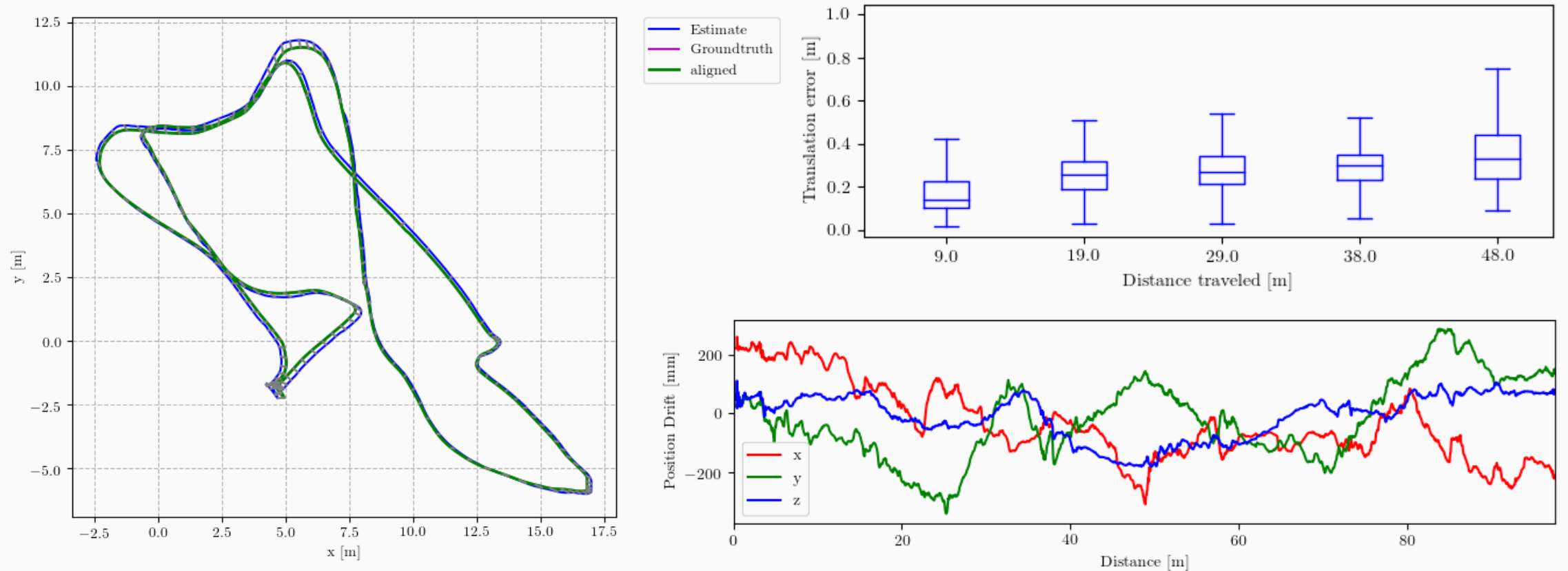
Advantage of RPE is that it captures **local trajectory errors**, making it particularly useful for evaluating odometry algorithms. Commonly, it separately evaluates translational and rotational errors.

Drawback is that we do not end up with a single number right away but have to decide on how to aggregate the statistics based on which algorithms can be compared.

Commonly, the average of all sub-trajectory errors (translation and rotation separately) is computed, where the lengths are either fixed beforehand or defined in percentages of the total trajectory length.

Relative pose error

An example of how an open-source package [rpg_trajectory_evaluation](#) computes and visualizes the odometry performance is depicted below.








Popular odometry datasets

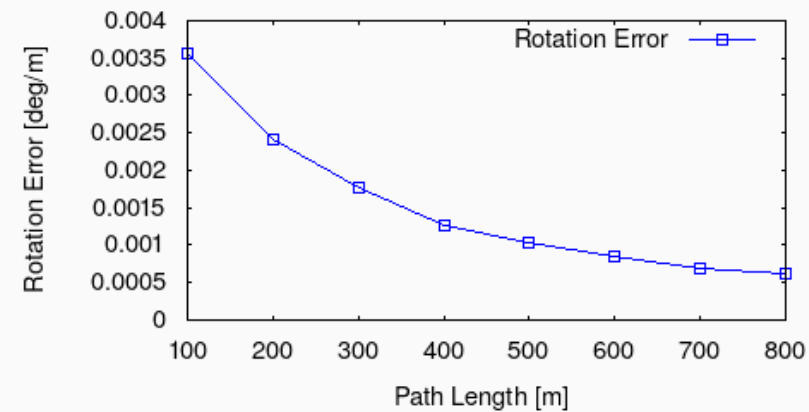
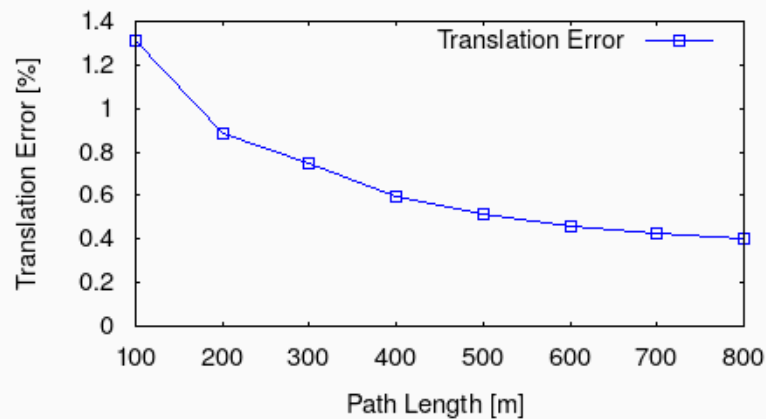
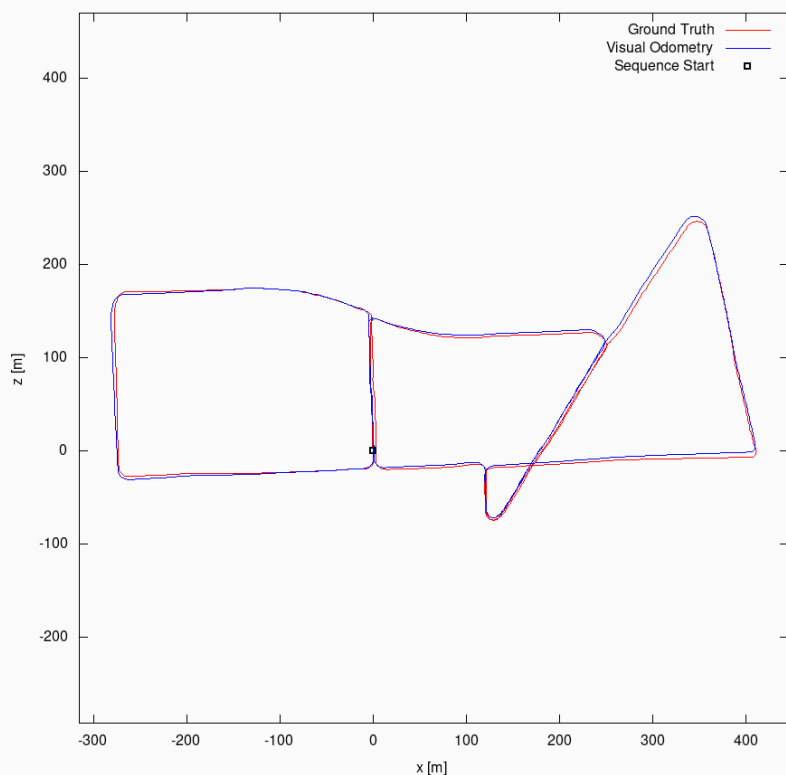
Researchers have devoted considerable effort in creating datasets that can be used for (visual) odometry evaluation. Many contain multiple sensor modalities and varying degree of ground truth availability and accuracy.

Most popular datasets are [Malaga Urban Dataset](#), [Multivehicle stereo event camera dataset](#), DSEC, [EuROC micro-aerial vehicle dataset](#), [Oxford RobotCar dataset](#), [KITTI](#), [KITTI-360](#), and [TUM datasets](#).

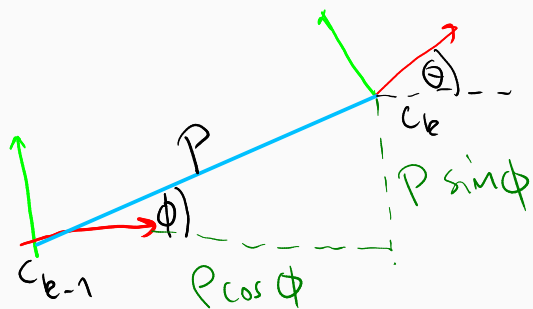
Among these special mention deserve KITTI and KITTI-360 as besides the dataset they also curate an online public benchmark where researchers can upload and evaluate their solutions on the test sequences for which ground truth data is not available.

KITTI evaluation example

	Method	Setting	Code	Translation	Rotation	Runtime	Environment	Compare
1	SOFT2			0.53 %	0.0009 [deg/m]	0.1 s	4 cores @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
<p>I. Cvišić, I. Marković and I. Petrović: SOFT2: Stereo Visual Odometry for Road Vehicles Based on a Point-to-Epipolar-Line Metric. IEEE Transactions on Robotics 2022. I. Cvišić, I. Marković and I. Petrović: Enhanced calibration of camera setups for high-performance visual odometry. Robotics and Autonomous Systems 2022. I. Cvišić, I. Marković and I. Petrović: Recalibrating the KITTI Dataset Camera Setup for Improved Odometry Accuracy. European Conference on Mobile Robots (ECMR) 2021.</p>								
2	V-LOAM			0.54 %	0.0013 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
<p>J. Zhang and S. Singh: Visual-lidar Odometry and Mapping: Low drift, Robust, and Fast. IEEE International Conference on Robotics and Automation(ICRA) 2015.</p>								
3	LOAM			0.55 %	0.0013 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
<p>J. Zhang and S. Singh: LOAM: Lidar Odometry and Mapping in Real-time. Robotics: Science and Systems Conference (RSS) 2014.</p>								
4	TVL-SLAM+	 		0.56 %	0.0015 [deg/m]	0.3 s	1 core @ 3.0 Ghz (C/C++)	<input type="checkbox"/>
<p>C. Chou and C. Chou: Efficient and Accurate Tightly-Coupled Visual-Lidar SLAM. IEEE Transactions on Intelligent Transportation Systems 2021.</p>								



The Board



$$E = [t]_x R$$

$$R = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$t = \begin{bmatrix} P \cos \phi \\ P \sin \phi \\ 0 \end{bmatrix}$$

$$[t]_x = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & P \sin \phi \\ 0 & 0 & -P \cos \phi \\ -P \sin \phi & P \cos \phi & 0 \end{bmatrix}$$

$\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$

$$E \approx \begin{bmatrix} 0 & 0 & P \sin \phi \\ 0 & 0 & -P \cos \phi \\ -P \sin \phi & P \cos \phi & 0 \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & P \sin \phi \\ 0 & 0 & -P \cos \phi \\ -P \sin \phi \cos \theta + P \cos \phi \sin \theta & P \sin \phi \sin \theta + P \cos \phi \cos \theta & 0 \end{bmatrix}$$

$- \sin(\phi - \theta)$ $P \cos(\phi - \theta)$

$$= \begin{bmatrix} 0 & 0 & P \sin \phi \\ 0 & 0 & -P \cos \phi \\ -\sin(\phi - \theta) & P \cos(\phi - \theta) & 0 \end{bmatrix} \left. \begin{array}{l} P \text{ is a} \\ \text{scale} \\ \text{factor} \\ (P = 1) \end{array} \right\}$$

θ_1 θ_2

$\rightarrow \phi, \theta$ 2 DoF

* 2 correspondences will suffice!

$$E = \begin{bmatrix} 0 & 0 & \sin \theta_2 \\ 0 & 0 & -\cos \theta_2 \\ -\sin \theta_1 & \cos \theta_1 & 0 \end{bmatrix}; \quad x'^T E x = 0$$

$$x = [x_1 \ y_1 \ z_1]^T, \quad x' = [x_2 \ y_2 \ z_2]^T$$

$$[x_2 \ y_2 \ 1] E \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \text{epipolar constraint}$$

$$= [-\sin \theta_1 \quad \cos \theta_1 \quad x_2 \sin \theta_2 - y_2 \cos \theta_2] \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

$$= -x_1 \sin \theta_1 + y_1 \cos \theta_1 + x_2 \sin \theta_2 - y_2 \cos \theta_2 = 0$$

unknowns

$$A x = 0 \Rightarrow x = \begin{bmatrix} \cos \theta_1 \\ \sin \theta_1 \\ \cos \theta_2 \\ \sin \theta_2 \end{bmatrix}, \quad A = [-y_1 \ y_1 \ x_2 \ -y_2]$$

* linear solution w/ 4 correspondences

* x elements are not independent:

$$\begin{aligned} \sin^2 \theta_1 + \cos^2 \theta_1 &= 1 \\ \sin^2 \theta_2 + \cos^2 \theta_2 &= 1 \end{aligned}$$

* if $u = \sin \theta_1$, $v = \sin \theta_2 \Rightarrow \cos \theta_1 = \sqrt{1-u^2}$
 $\cos \theta_2 = \sqrt{1-v^2}$

$$-x_1 u + x_2 v + y_1 \sqrt{1-u^2} - y_2 \sqrt{1-v^2} = 0 \quad \leftarrow 1 \text{ pt}$$

$\leftarrow 2 \text{ pts}$

* epipolar constraint for 1 point $p^T E p^T$:

$$\begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & \rho \sin \frac{\theta}{2} \\ 0 & 0 & \rho \cos \frac{\theta}{2} \\ \rho \sin \frac{\theta}{2} & \rho \cos \frac{\theta}{2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

↑ scale factor ($\rho=1$)

$$= x_1 \sin \frac{\theta}{2} + y_1 \cos \frac{\theta}{2} + x_2 \sin \frac{\theta}{2} - y_2 \cos \frac{\theta}{2} = 0$$

$$= \sin \frac{\theta}{2} (x_1 + x_2) - \cos \frac{\theta}{2} (y_2 - y_1) = 0$$

$$\Rightarrow \tan \frac{\theta}{2} = \frac{y_2 - y_1}{x_1 + x_2}$$

$$\theta = 2 \tan^{-1} \frac{y_2 - y_1}{x_1 + x_2}$$