Low-level Vision Tasks

Visual Odometry

Kuk-Jin Yoon

Visual Intelligence Lab.
Department of Mechanical Engineering

KAIST

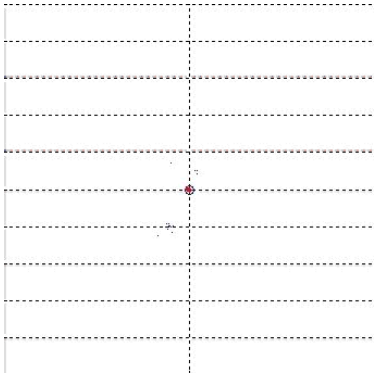
What is Visual Odometry (VO) ?

- VO is the process of incrementally estimating the pose of the vehicle by examining the changes that motion induces on the images of its onboard cameras

input

Image sequence (or video stream)
from one or more cameras attached to a moving vehicle

output



$R_0, R_1, \ldots, R_i$

$t_0, t_1, \ldots, t_i$

Camera trajectory (3D structure is a plus):

Basic Assumptions on the Working Environments

- Sufficient illumination in the environment
- Dominance of static scene over moving objects
- Enough texture to allow apparent motion to be extracted
- Sufficient scene overlap between consecutive frames

KAIST

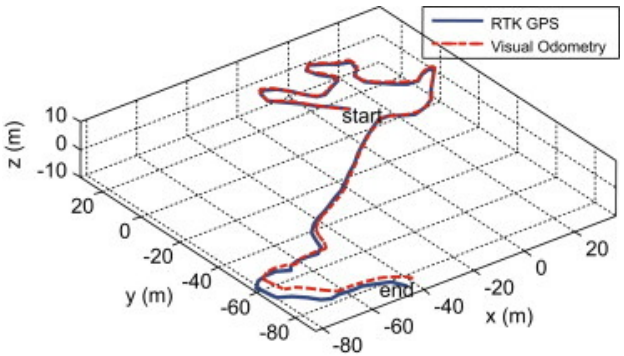
Why VO ?

- Contrary to wheel odometry, VO is not affected by wheel slip in uneven terrain or other adverse conditions.

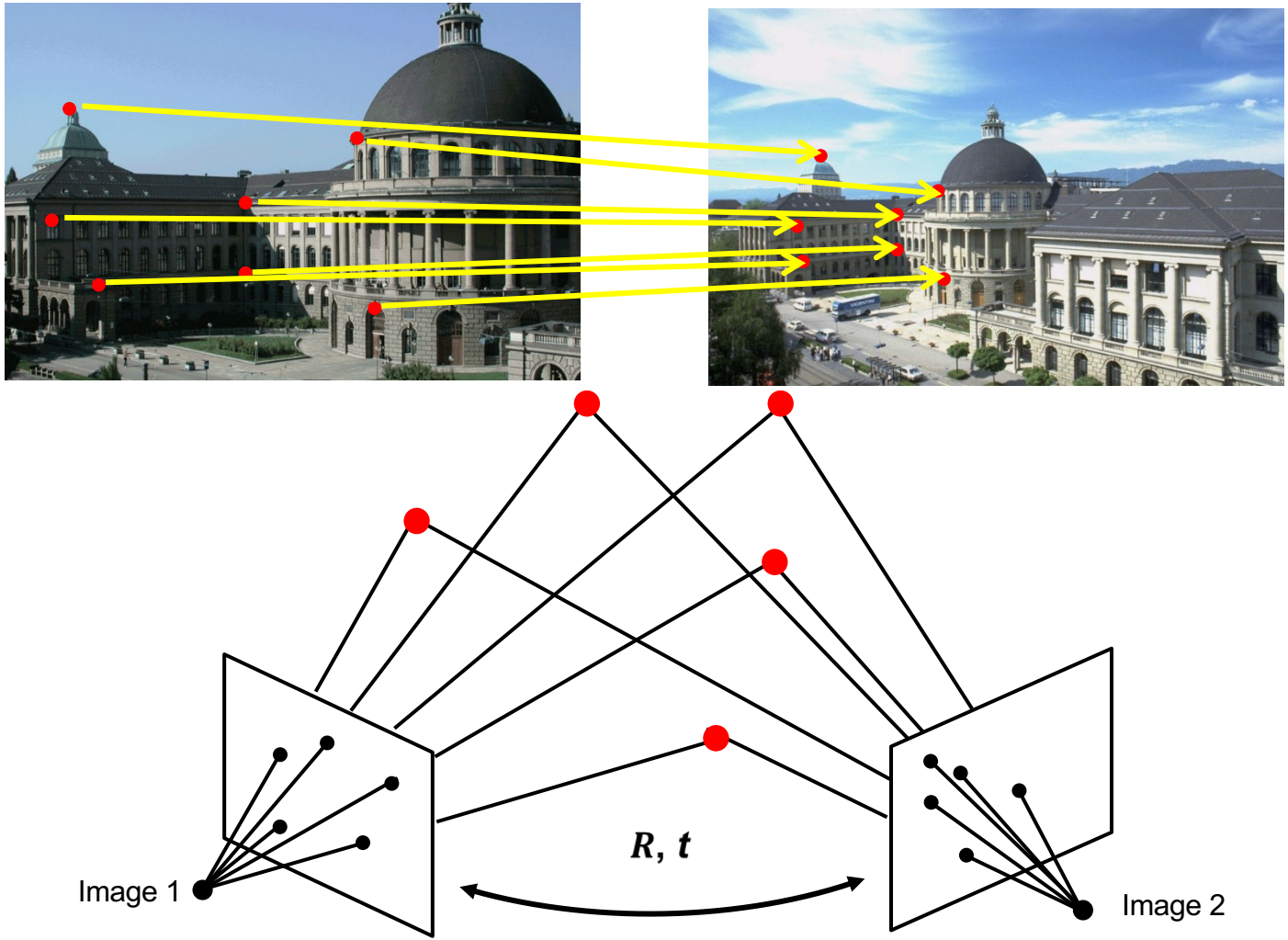
- More accurate trajectory estimates compared to wheel odometry
(relative position error 0.1% – 2%)

- VO can be used as a complement to
  - wheel odometry
  - GPS
  - inertial measurement units (IMUs)
  - laser odometry

- In GPS-denied environments, such as underwater and aerial, VO has utmost importance.

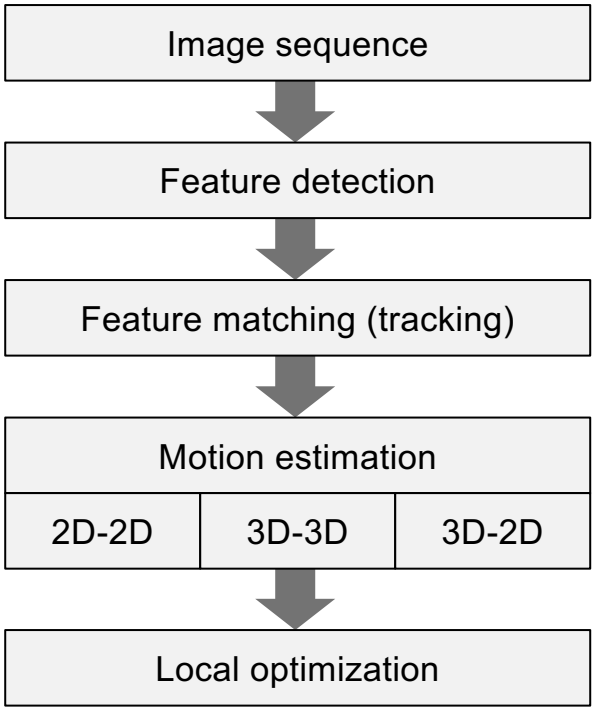


Working Principle

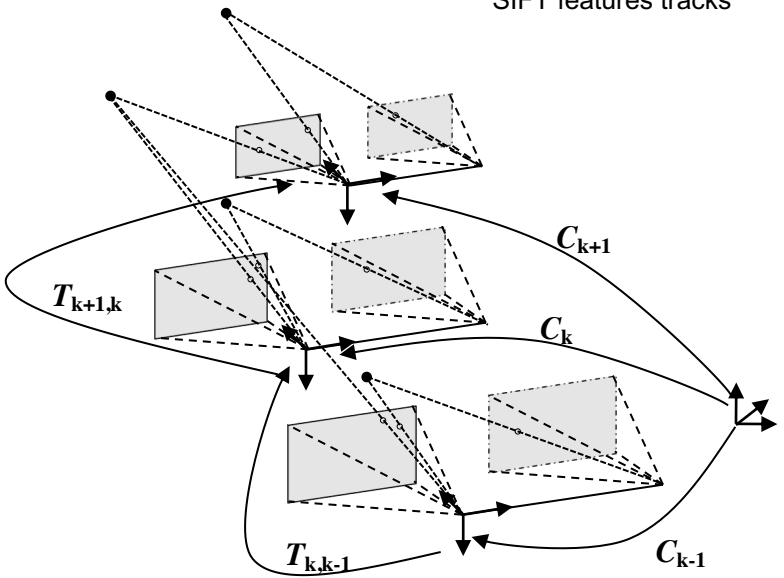


VO Flow Chart

- VO computes the camera path incrementally (pose after pose)

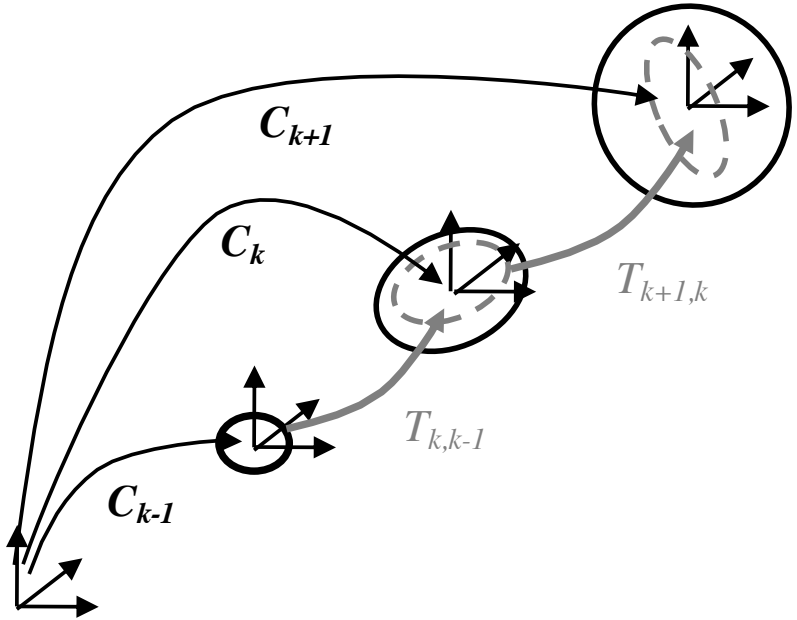


SIFT features tracks



VO Drift

- The errors introduced by each new frame-to-frame motion accumulate over time.
- This generates a drift of the estimated trajectory from the real path



The uncertainty of the camera pose at C_k is a combination of the uncertainty at C_{k-1} (black solid ellipse) and the uncertainty of the transformation $T_{k,k-1}$ (gray dashed ellipse)

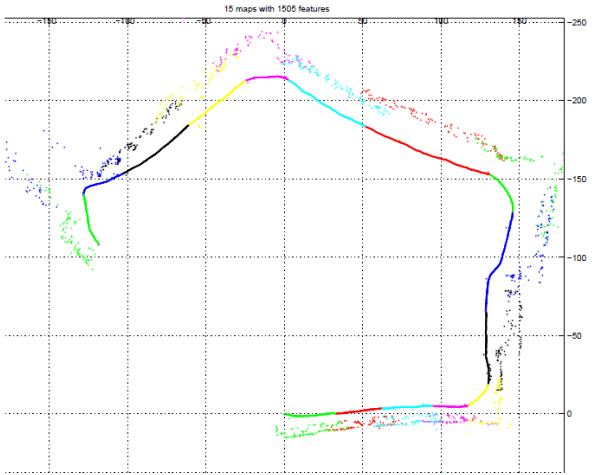
VO versus Structure from Motion (SFM)

- SFM is more general than VO and tackles the problem of 3D reconstruction of both the structure and camera poses from unordered image sets
- The final structure and camera poses are typically refined with an offline optimization (i.e., bundle adjustment), whose computation time grows with the number of images
- VO is a particular case of SFM
- VO focuses on estimating the 3D motion of the camera sequentially (as a new frame arrives) and in real time
- Bundle adjustment can be used (but it's optional) to refine the local estimate of the trajectory
- Terminology: sometimes SFM is used as a synonym of VO

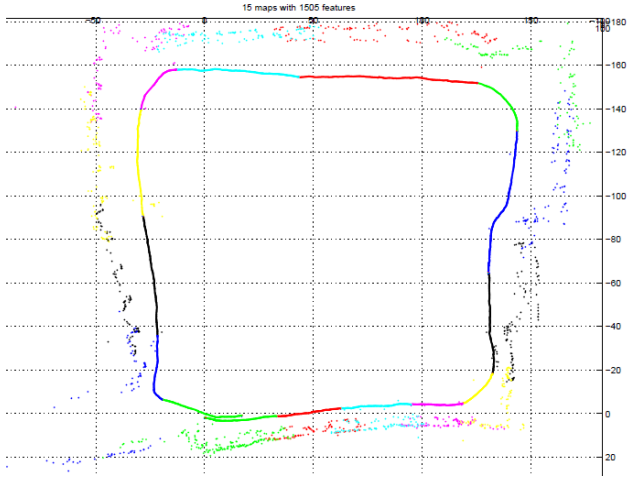
VO versus Visual SLAM

- The goal of SLAM in general is to obtain a global, consistent estimate of the robot path. This is done through identifying loop closures. When a loop closure is detected, this information is used to reduce the drift in both the map and camera path (global bundle adjustment).

- Conversely, VO aims at recovering the path incrementally, pose after pose, and potentially optimizing over the last a few poses path (local or windowed bundle adjustment)



before loop closing

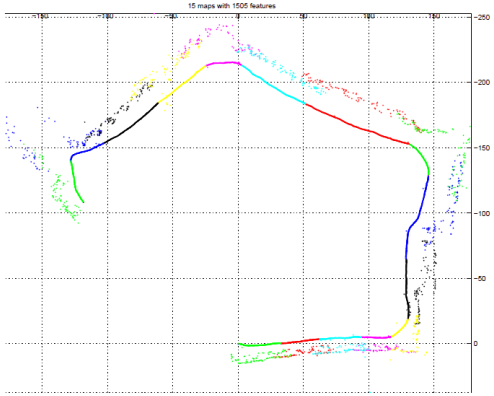


after loop closing

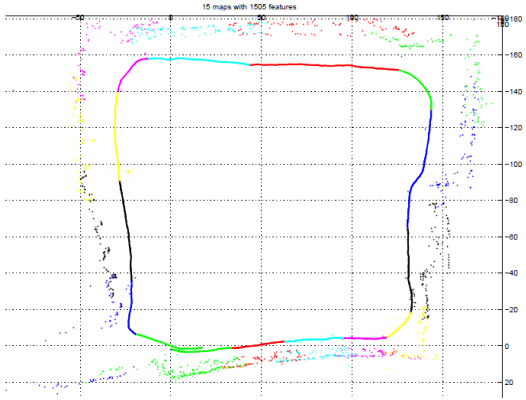
Image courtesy of Clemente et al. RSS'07

VO versus Visual SLAM

- VO only aims to the local consistency of the trajectory
- SLAM aims to the global consistency of the trajectory and of the map
- VO can be used as a building block of SLAM
- VO is SLAM before closing the loop!
- The choice between VO and V-SLAM depends on the tradeoff between performance and consistency, and simplicity in implementation
- VO trades off consistency for real-time performance, without the need to keep track of all the previous history of the camera



Visual odometry



Visual SLAM

Image courtesy of Clemente et al. RSS'07

Problem Formulation

- An agent is moving through an environment and taking images with a rigidly-attached camera system at discrete times k

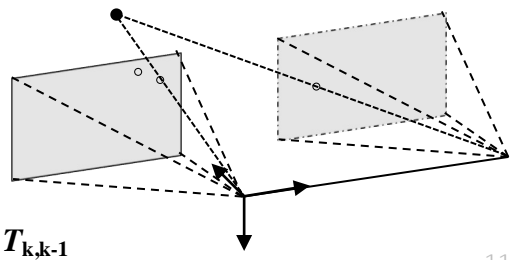
- In case of a monocular system, the set of images taken at times k is denoted by

$$I_{0:n} = \{I_0, \dots, I_n\}$$

- In case of a stereo system, there are a left and a right image at every time instant, denoted by

$$I_{l,0:n} = \{I_{l,0}, \dots, I_{l,n}\},$$
$$I_{r,0:n} = \{I_{r,0}, \dots, I_{r,n}\}$$

- In case of a stereo system, without loss of generality, the coordinate system of the left camera can be used as the origin



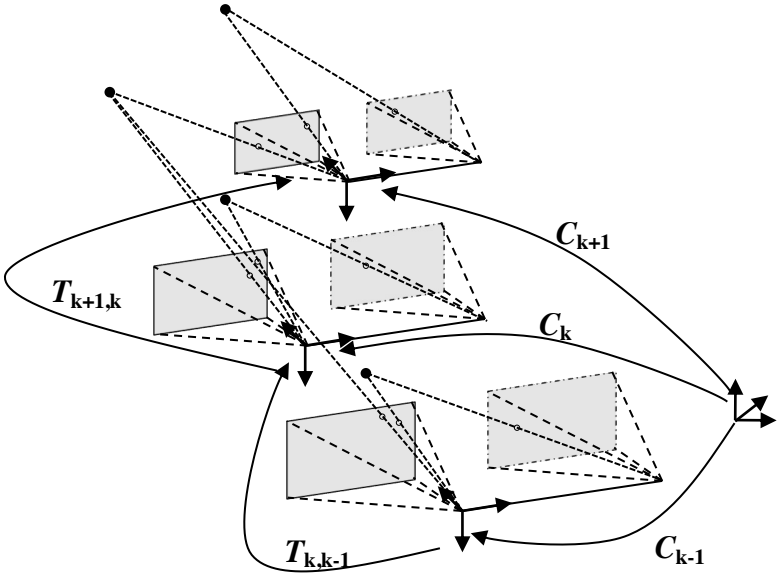
Problem Formulation

- Two camera positions at adjacent time instants $k-1$ and k are related by the rigid body transformation

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}$$

- The set $T_{0:n} = \{T_1, \dots, T_n\}$ contains all subsequent motions all subsequent motions.

- Finally, the set of camera poses $C_{0:n} = \{C_0, \dots, C_n\}$ contains the transformations of the camera with respect to the initial coordinate frame at $k = 0$

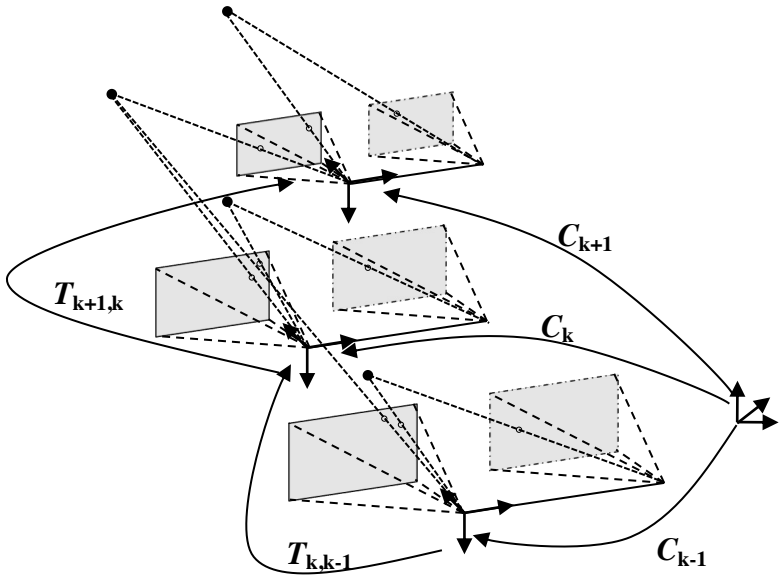


Problem Formulation

- The current pose C_n can be computed by concatenating all the transformations T_k, $k = 1 \ldots n$, and, therefore,

$$C_n = C_{n-1} T_n$$

with C_0 being the camera pose at the instant $k = 0$, which can be set arbitrarily by the user

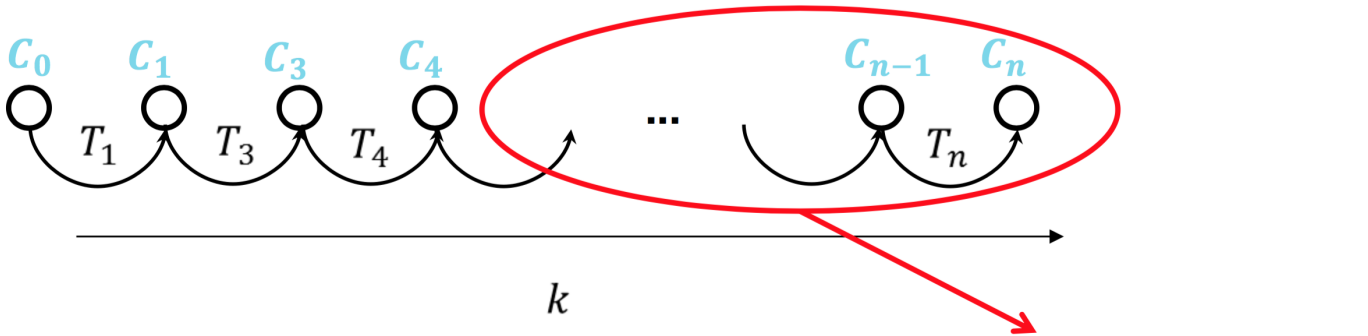


VO Problem Formulation

- Computing the transformation between two consecutive frames and concatenate the all transformations to recover the full trajectory of a camera

- This means that VO recovers the path incrementally, pose after pose.

- An iterative refinement over last *m* poses can be performed after this step to obtain a more accurate estimate of the local trajectory

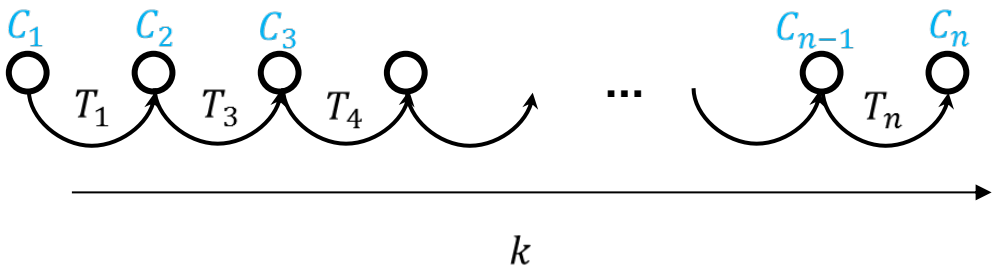


Motion Estimation

- Motion estimation is the core computation step performed for every image in a VO system
- It computes the camera motion T_k between the previous and the current image:

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}$$

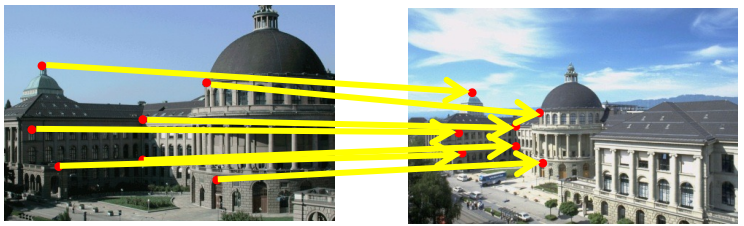
- By concatenation of all these single movements, the full trajectory of the camera can be recovered



Appearance-based or Featured-based ?

There are two main approaches to compute the relative motion T_k

- Appearance-based methods use the intensity information of all the pixels in the two input images
- Feature-based methods only use salient and repeatable features extracted (or tracked) across the images



Makadia et al. «Correspondence-free structure from motion», IJCV'07

Appearance-based or Featured-based ?

- Global methods are less accurate than feature-based methods and are computationally more expensive.

- Feature-based methods require the ability to match (or track) robustly features across frames but are faster and more accurate than global methods. Therefore, most VO implementations are feature based.

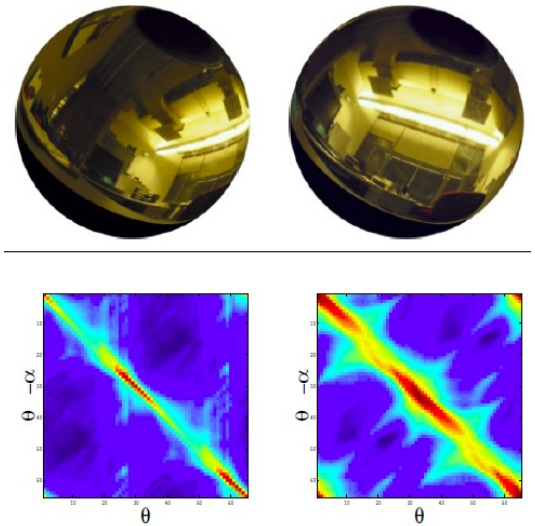
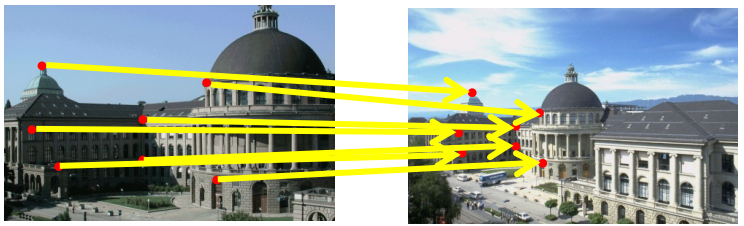


Image courtesy of Makadia et al., IJCV'07



Motion Estimation

Motion estimation		
2D-2D	3D-3D	3D-2D

- Depending on whether the feature correspondences f_{k-1} and f_k are specified in 2D or 3D, there are three different cases:

- **2D-to-2D**: both f_{k-1} and f_k are specified in 2D image coordinates

- **3D-to-3D**: both f_{k-1} and f_k are specified in 3D To do this, it is necessary to triangulate 3D points at each time instant, for instance, by using a stereo camera system

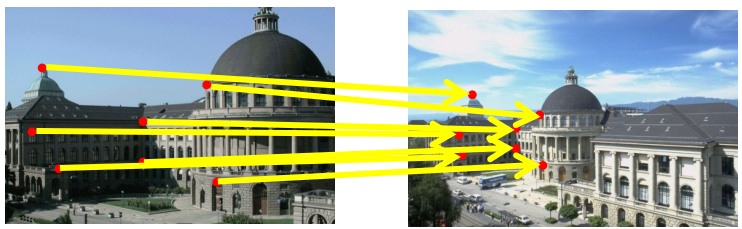
- **3D-to-2D**: f_{k-1} are specified in 3D and f_k are their corresponding 2D reprojections on the image I_k

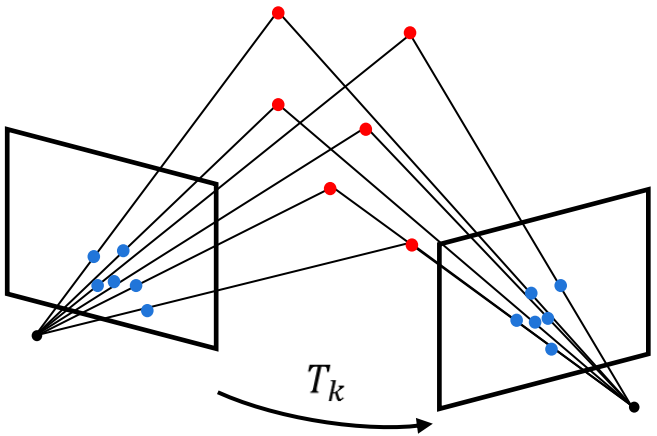
2D-to-2D: Motion from Image Feature Correspondences

Motion estimation		
2D-2D	3D-3D	3D-2D

- Both f_{k-1} and f_k are specified in 2D
- The minimal-case solution involves 5-point correspondences
- The solution is found by determining the transformation that minimizes the reprojection error of the triangulated points in each image

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{X^i, C_k} \sum_{i,k} \| p_k^i - g(X^i, C_k) \|^2$$

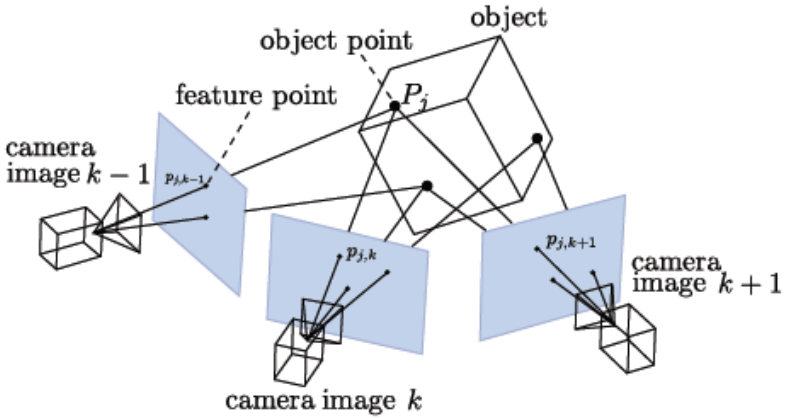




VO for Two Frame

Motion estimation		
2D-2D	3D-3D	3D-2D

- Compute essential matrix $\mathbf{E}$ between the two views
  - First camera matrix: **[I|0]**
  - Second camera matrix: **[R|t]**



$$z\mathbf{x} = [\mathbf{I}\,|\,\mathbf{0}]\mathbf{X}, \quad z'\mathbf{x}' = [\mathbf{R}\,|\,\mathbf{t}]\mathbf{X}$$

$$\Rightarrow \quad z'\mathbf{x}' = \mathbf{R}[\mathbf{I}\,|\,\mathbf{0}]\mathbf{X} + \mathbf{t} = z\mathbf{R}\mathbf{x} + \mathbf{t}$$

$$\Rightarrow \quad \mathbf{t} \times (z'\mathbf{x}') = \mathbf{t} \times (z\mathbf{R}\mathbf{x})$$

$$\Rightarrow \quad \mathbf{x}' \cdot (\mathbf{t} \times (z'\mathbf{x}')) = \mathbf{x}' \cdot (\mathbf{t} \times (z\mathbf{R}\mathbf{x})) = 0$$

$$\mathbf{x}'^{\mathrm{T}}[\mathbf{t}_\times]\mathbf{R}\mathbf{x} = 0$$

$$\mathbf{E} = [\mathbf{t}_\times]\mathbf{R}$$

Pose from the Essential Matrix

- Taking the SVD of the essential matrix, and then exploiting the constraints on the rotation matrix, we get the following:

$$E = U\Sigma V^T$$

- The translation and axis and angle of rotation can then be obtained directly up to arbitrary signs and *unknown* scale for the translation:

$$[\mathbf{T}]_\times = \mathbf{U}\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{U}^\top$$

$$\mathbf{R} = \mathbf{U}\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\mathbf{V}^\top$$

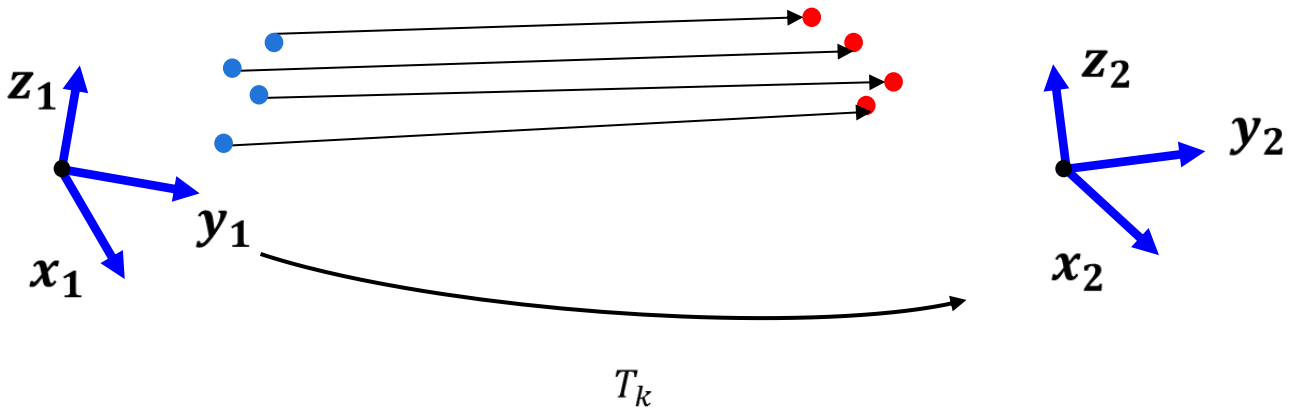
Four solutions are still possible due to the arbitrary choice of signs for the translation T and rotation R, however the correct one is easily disambiguated by ensuring that the reconstructed points lie in front of the cameras.

3D-to-3D: Motion from 3D-3D Point Correspondences

Motion estimation		
2D-2D	3D-3D	3D-2D

- Both f_{k-1} and f_k are specified in 3D
- To do this, it is necessary to triangulate 3D points (e.g. use a stereo camera)
- The minimal-case solution involves 3 non-collinear correspondences
- The solution is found by determining the aligning transformation that minimizes the 3D-3D distance

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{T_k} \sum_i ||\tilde{X}_k^i - T_k \tilde{X}_{k-1}^i||$$

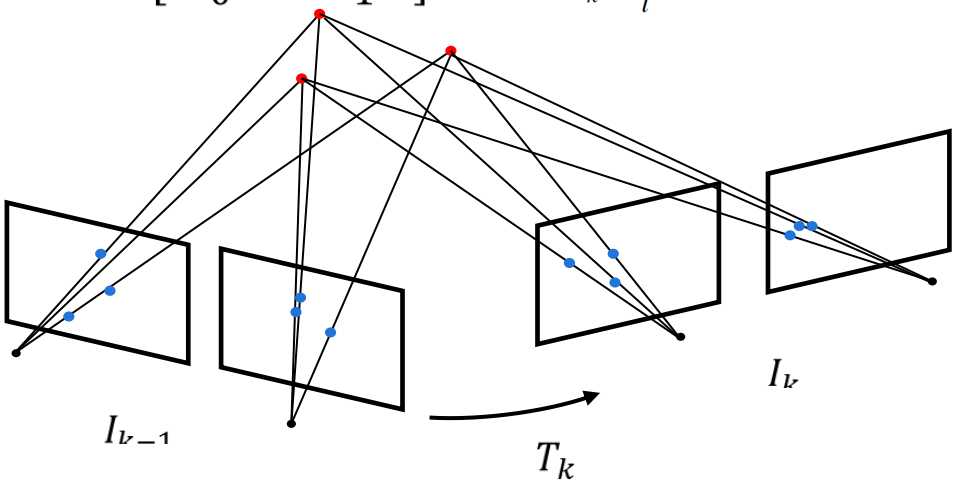


3D-to-3D: Motion from 3D-3D Point Correspondences

Motion estimation		
2D-2D	3D-3D	3D-2D

- Both f_{k-1} and f_k are specified in 3D
- To do this, it is necessary to triangulate 3D points (e.g. use a stereo camera)
- The minimal-case solution involves 3 non-collinear correspondences
- The solution is found by determining the aligning transformation that minimizes the 3D-3D distance

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{T_k} \sum_i ||\tilde{X}_k^i - T_k \tilde{X}_{k-1}^i||$$

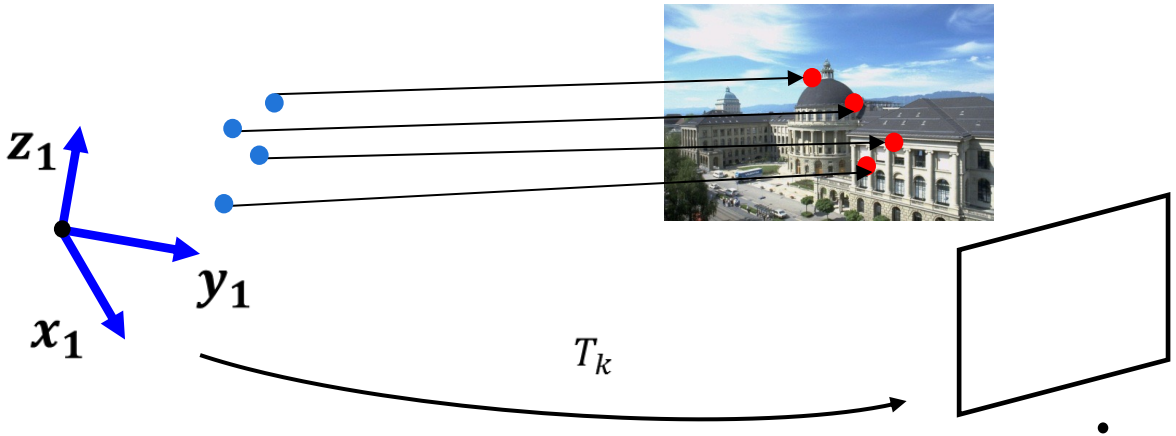


3D-to-2D: Motion from 3D Structure and Image Correspondences

Motion estimation		
2D-2D	3D-3D	3D-2D

- f_{k-1} is specified in 3D and f_k in 2D
- This problem is known as camera resection or PnP(perspective from n points)
- The minimal-case solution involves 3 correspondences
- The solution is found by determining the transformation that minimizes the reprojection error

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{T_k} \sum_i \| p_k^i - \hat{p}_{k-1}^i \|^2$$

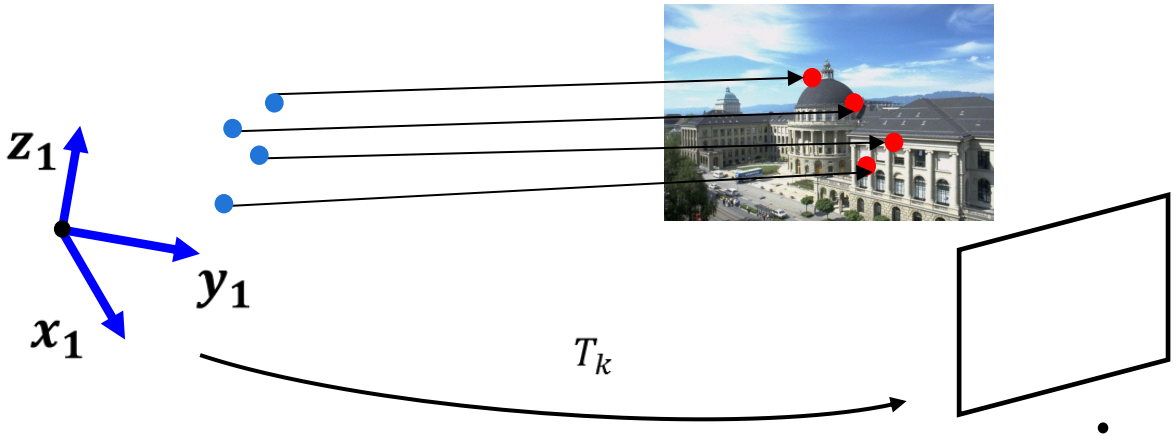


3D-to-2D: Motion from 3D Structure and Image Correspondences

Motion estimation		
2D-2D	3D-3D	3D-2D

- In the monocular case, the 3D structure needs to be triangulated from two adjacent camera views (e.g., I_{k-2} and I_{k-1}) and then matched to 2D image features in a third view (e.g., I_k).

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{T_k} \sum_i \| p_k^i - \hat{p}_{k-1}^i \|^2$$



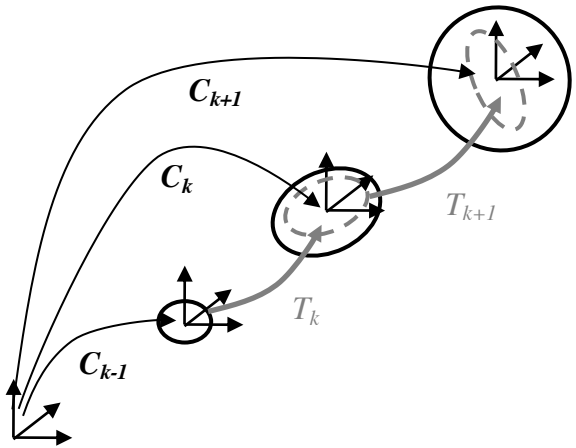
Error Propagation

- The uncertainty of the camera pose C_k is a combination of the uncertainty at C_{k-1} (black-solid ellipse) and the uncertainty of the transformation T_k (gray dashed ellipse)

- $C_k = f(C_{k-1}, T_k)$

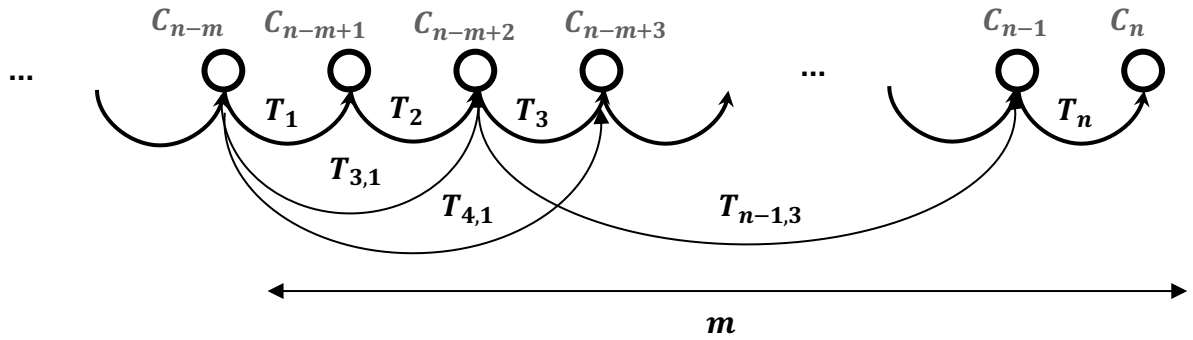
- The combined covariance $\sum_k$ is

$$\begin{aligned}\Sigma_k &= J \begin{bmatrix} \Sigma_{k-1} & 0 \\ 0 & \Sigma_{k,k-1} \end{bmatrix} J^\top \\ &= J_{\vec{C}_{k-1}} \Sigma_{k-1} J_{\vec{C}_{k-1}}{}^\top + J_{\vec{T}_{k,k-1}} \Sigma_{k,k-1} J_{\vec{T}_{k,k-1}}{}^\top\end{aligned}$$



- The camera-pose uncertainty is always increasing when concatenating transformations. Thus, it is important to keep the uncertainties of the individual transformations small.

Windowed Camera-Pose Optimization

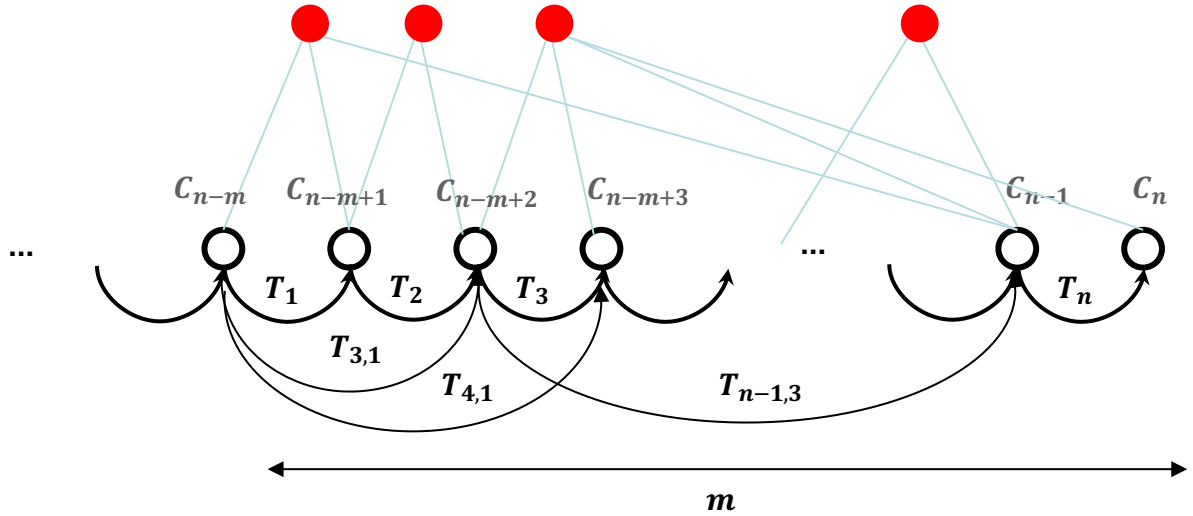


- So far we assumed that the transformations are between consecutive frames
- Transformations can be computed also between non-adjacent frames $T_{e_{ij}}$ and can be used as additional constraints to improve cameras poses by minimizing the following

$$\sum_{e_{ij}} \|C_i - T_{e_{ij}} C_j\|^2$$

- For efficiency, only the last m key frames are used.
- Levenberg-Marquadt can be used.

Windowed Bundle Adjustment (BA)



- Similar to pose-optimization but it also optimizes 3D points

$$\arg\min_{X^i, C_k} \sum_{i,k} \| p_k^i - g(X^i, C_k) \|^2$$

- In order to not get stuck in local minima, the initialization should be close the minimum
- Levenberg-Marquadt can be used

Considerations about Bundle Adjustment

- Windowed BA reduces the drift compared to 2-view VO because incorporates constraints between several frames
- More precise than camera-pose optimization
- The choise of the window size m is governed by computational reasons
- The computational complexity of BA is $O\left((qN + lm)^3\right)$ with N being the number of points, m the number of poses, and q and m the number of parameters for points and camera poses

Loop Detection

- Loop constraints are very valuable constraints for pose graph optimization
- These constraints form graph edges between nodes that are usually far apart and between which large drift might have been accumulated.
- Events like reobserving a landmark after not seeing it for a long time or coming back to a previously-mapped area are called loop detections
- Loop constraints can be found by evaluating visual similarity between the current camera images and past camera images.
- Visual similarity can be computed using global image descriptors or local image descriptors

First observation

Second observation after a loop

Image courtesy of Cummins & Newman, IJRR'08

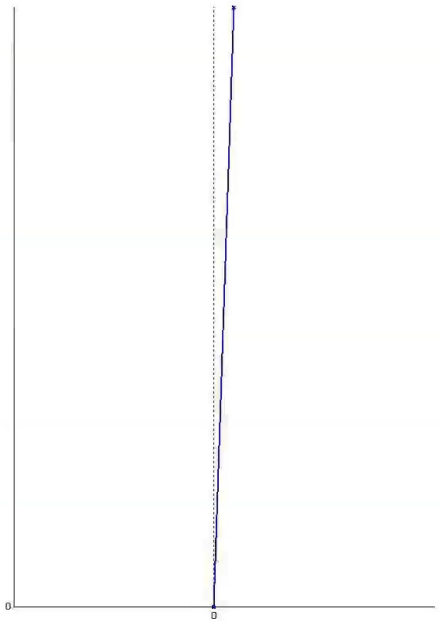
Improving the Accuracy of VO

- Other sensors can be used such as
  - IMU (called inertial VO)
  - Compass
  - GPS
  - Laser
- An IMU combined with a single camera allows the estimation of the absolute scale.
- Make sure that you have many points (thoudsands) which cover the image uniformly

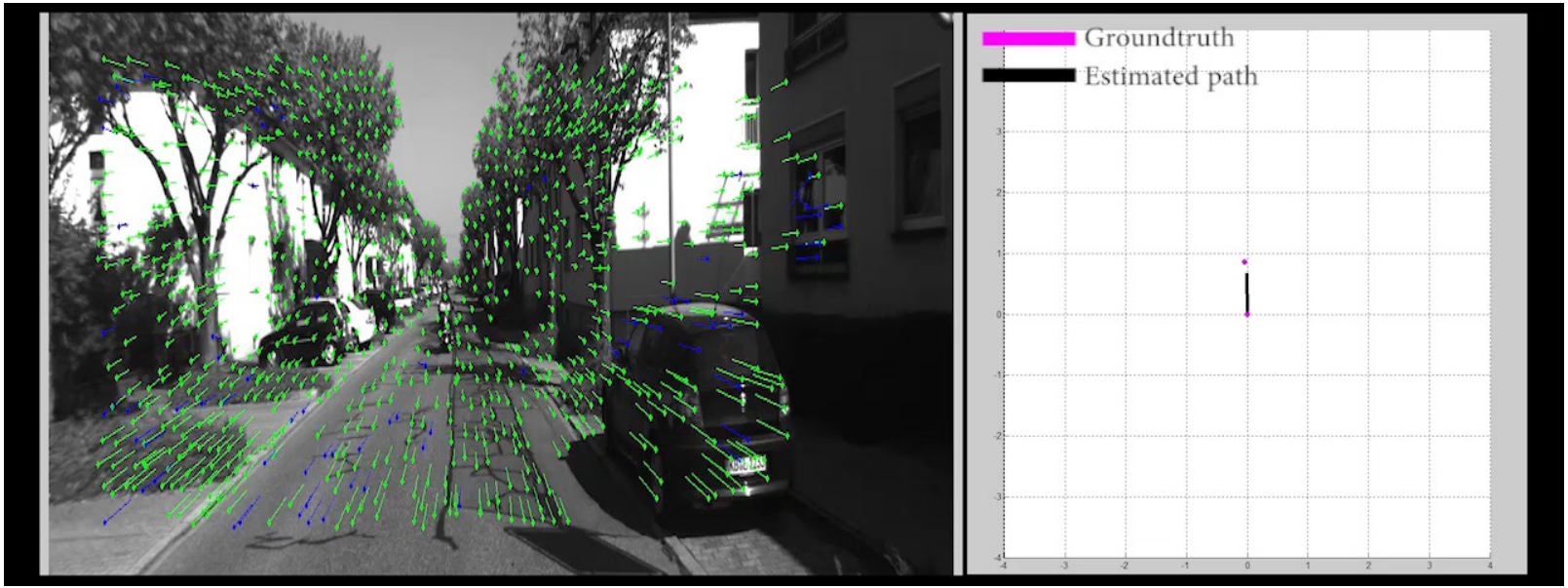
Mono / Stereo VO

- In the stereo vision case, 3D-2D method exhibits less drift than 3D-3D method.
- Stereo vision has the advantage over monocular vision that both motion and structure are computed in the absolute scale. It also exhibits less drift.
- When the distance to the scene is much larger than the stereo baseline, stereo VO degenerates into monocular VO.
- Keyframes should be selected carefully to reduce drift.
- Regardless of the chosen motion computation method, local bundle adjustment (over the last m frames) should be always performed to compute a more accurate estimate of the trajectory. After bundle adjustment, the effects of the motion estimation method are much more alleviated (as long as the initialization is close to the solution).

Demo



Demo



Mono SLAM Algorithm Outline

- Capture images I(k), I(k+1)
- Undistort the above images (optional)
- Detect features in I(k), and track those features to I(k+1).
  - A new detection is triggered if the number of features drop below a certain threshold.
- Use Nister's 5-point algorithm with RANSAC to compute the essential matrix
- Estimate **R** and **t** from the essential matrix that was computed in the previous step
- Take scale information from some external source (like a speedometer), and concatenate the translation vectors, and rotation matrices.